DOCUMENT RESUME

ED 209 889 EC 140 606

AUTHOR	Nazzaro, Jean N., Ed.
TITLE	Computer Connections for Gifted Children and Youth.
INSTITUTION	ERIC Clearinghouse on Handicapped and Gifted Children, Reston, Va.
SPONS AGENCY	National Inst. of Education (ED), Washington, D.C.
REPORT NO	ISBN-0-86586-119-6
PUB DATE	81
GRANT	400-76-0119
NOTE	87p.
AVAILABLE FROM	ERIC Clearinghouse on Handicapped and Gifted Children, The Council for Exceptional Children, 1920 Association Dr., Reston, VA 22091 ($1.00).
EDRS PRICE	MF01/PC04 Plus Postage.
DESCRIPTORS	*Computer Assisted Instruction; *Computers; Elementary Secondary Education; *Gifted; *Microcomputers; Program Descriptions; Programing; *Talent

ABSTRACT

Written by computer specialists, teachers, parents, and students, the 23 articles emphasize the role computers play in the development of thinking, problem solving, and creativity in gifted and talented students. Articles have the following titles and authors: "Computers and Computer Cultures" (S. Papert); "Classroom Computers--Beyond the 3 R's" (P. Bell); "Reflections of a Computer Language Nut" (S. Bloch); "It Started with Games" (C. Karnes); "Two Programs from a Young Eighth Grader" (S. Bahcall, H. Nelson); "Teaching Parents About Using Microcomputers" (T. Dwyer, M. Critchfield); "Children and Home Computers--Some Observations on the First Generation" (B. Banet); "An Apple a Day Keeps a Kid Occupied" (R. Buszta); "Microcomputers for Gifted Microtots" (A. Doorly); "Kids and Computers--The Future Is Today" (S. Larsen); "Micros 'GOTO' School" (D. Piele); "The Hampton City Schools Computer Program" (N. Harkavy); "The Paducah Tilghman High School for Gifted Students" (S. Davis, P. Frothingham); "Computers....Are All Dinosaurs Dead?" (D. Glover); "The Talcott Mountain Science Center" (D. Barstow); "Computeronics--A Course in Computer Literacy" (P. Bird); "Bringing Microcomputers into Schools" (G. Ropes, H. Gaylord); "Statewide Educational Computer Systems--The Many Considerations" (K. Hausmann); "Van Helps Schools Select the Right Computer" (B. Staples); "Some Thoughts on Computers and Greatness in Teaching" (T. Dwyer); "The Hacker Papers;" and "Educational Software" (D. Lubar). Lists of printed materials, vendors and manufacturers, and resources on using microcomputers in schools and classrooms conclude the document. (SB)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED209889

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION
CENTER (ERIC)

☑ This document has been reproduced as received from the person or organization originating it.
☐ Minor changes have been made to improve reproduction quality.

- Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

COMPUTER CONNECTIONS

for Gifted Children and Youth

JEAN N. NAZZARO, Editor

EC 140 606

The Council for Exceptional Children

A product of the
ERIC Clearinghouse
on Handicapped and Gifted Children

Cover photograph by Gerald A. Tate

A product of the ERIC Clearinghouse on Handicapped and Gifted Children.

Published in 1981 by The Council for Exceptional Children, 1920 Association Drive, Reston, Virginia 22091

Library of Congress Number 81–69423

ISBN 0–86586–119–6

This publication was prepared with funding from the National Institute of Education, US Department of Education under contract no. 400–76–0119. Contractors undertaking such projects under government sponsorship are encouraged to express freely their judgment in professional and technical matters. Prior to publication the manuscript was submitted to The Council for Exceptional Children for critical review and determination of professional competence. This publication has met such standards. Points of view, however, do not necessarily represent the official view or opinions of either The Council for Exceptional Children, the National Institute of Education, or the Department of Education.

Printed in the United States of America.

Contents

Foreword iv

1 Why Computer Experiences Are Needed Now

Computers and Computer Cultures ☐ Seymour Papert 3
Classroom Computers: Beyond the 3 R's ☐ Fred Bell 9

2 Perspectives of Gifted Children and Youth

Reflections of a Computer Language Nut ☐ Stephen Bloch 15
It Started with Games ☐ Christopher Karnes 18
Two Programs from a Young Eighth Grader ☐ Safi Bahcall and Harold Nelson 19

3 Computers at Home

Teaching Parents About Using Microcomputers ☐ Thomas A. Dwyer and Margot Critchfield 25
Children and Home Computers: Some Observations on the First Generation ☐ Bernard Banet 27
An Apple a Day Keeps a Kid Occupied ☐ Ron Buszta 28

4 Programs in Action

Microcomputers for Gifted Microtots ☐ Ann Doorly 31
Kids and Computers: The Future Is Today ☐ Sally Greenwood Larsen 34
Micros "GOTO" School ☐ Donald T. Piele 37
The Hampton City Schools Computer Program ☐ Nedra Harkavy 40
The Paducah Tilghman High School "Chemics" Program ☐ Steve Johnston 41
Computing at a New Public High School for Gifted Students ☐ Steve Davis and Phyllis S. Frothingham 43
Computers . . . Are All Dinosaurs Dead? ☐ Douglas Glover 46
The Talcott Mountain Science Center ☐ Daniel Barstow 49
Computeronics: A Course in Computer Literacy ☐ Pristen Bird 53

5 Getting Started

Bringing Microcomputers into Schools ☐ George Ropes and Henry Gaylord 57
Statewide Educational Computer Systems—The Many Considerations ☐ Kevin Hausmann 60
Van Helps Schools Select the Right Computer ☐ Betsy Staples 63

6 The Human Perspective

Some Thoughts on Computers and Greatness in Teaching ☐ Thomas A. Dwyer 71
The Hacker Papers 73

7 Resources

Educational Software ☐ David Lubar 83
Printed Materials 87
Vendors and Manufacturers 88
R & D Mini-List of Resources 90

Foreword

The launch of Sputnik in 1957 served to focus attention on our nation's need to provide special education for gifted and talented students. The event was like an electrical shock in that it served to jolt the education community into action with a resulting surge of programs, especially in science and mathematics. At that time, computers were very expensive, very large, and very limited in accessibility. The thought of being able to provide computer experiences to youngsters in regular classrooms was unthinkable.

Nearly 25 years has elapsed since Sputnik's flight and advancements in computer technology have turned what was once unthinkable into something that is becoming commonplace.

Although more computers are showing up in classrooms every year, there is still a long way to go before they will be universally available.

Competition among world powers is as keen today as it was 25 years ago. Although we haven't been jolted recently as we were with Sputnik there is no room for complacency. Most people agree that computer literacy will be essential for survival in the 21st century. Therefore, there is no time to waste. It is urgent that our schools provide computer experiences for today's students, especially those who are gifted.

The articles in this book clearly reveal that experiences with computers provide students with insight into the thinking process.

It is hoped that, after reading this book, educators and other individuals in decision-making positions will be motivated to do something about providing computer experiences for all students.

Articles written by computer specialists, teachers, parents, and students all emphasize the important role computers play in the development of thinking, problem solving, and creativity. Model program strategies for getting started and information on the most current resources have been included to help administrators who are ready to make the computer connection for their students.

My sincere thanks to Russell E. Adams who provided me with numerous articles for consideration from his personal collection of computer magazines. His advice and corroboration of my selections for this volume was greatly appreciated.

Jean N. Nazzaro
October 1981

1 Why Computer Experiences Are Needed Now

Computers and Computer Cultures by Seymour Papert and *Classroom Computers: Beyond the 3 R's* by Fred Bell were chosen as the first sections in this book because they so clearly articulate the value of computers beyond computer-assisted instruction. These articles emphasize that real learning comes about when students have the chance to experiment and program computers themselves.

Papert comments . . . "in teaching the computer how to think, children embark on an exploration about how they themselves think. Thinking about thinking turns the child into an epistemologist, an experience not even shared by most adults." If this happens with normal children, the insights gained by gifted youngsters may be even more profound.

Papert, a mathematician and a student of Piaget, believes that computer experiences for young children will have a profound effect on the age at which they are able to shift from concrete to formal thinking. Computer models provide concrete form to areas of knowledge that ordinarily are intangible and abstract. Furthermore, when children learn to program, the process of learning becomes active and self-directed. The knowledge is acquired for a recognizable personal purpose. Computers give students some real control over what they learn and how they learn it and this is very motivating.

Bell believes that computers can revolutionize learning in schools by teaching children how to function at higher cognitive levels. When a student writes computer programs, the six steps of problem solving (posing the problem, precisely defining the problem, gathering information, developing a solution strategy, finding the solution, and checking the solution) must be carried out. Children with these skills are capable of learning independently and of conducting the research and explorations that go into creating knowledge.

It seems quite evident that children who have been involved in the kind of experiences Papert and Bell describe will have distinct advantages over those who have not, not only because they will be computer literate in an age where computers have become indispensable tools, but because they will understand and be able to control their own thinking and problem-solving behavior.

Computers and Computer Cultures

Seymour Papert

This material is a condensation of Chapter 1 of **Mindstorms** *by Seymour Papert, published in 1980 by Basic Books, Inc., of New York City at $12.95. Those who find themselves stimulated by this presentation will find it profitable to read the entire work.*

In most contemporary educational situations where children come into contact with computers the computer is used to put children through their paces, to provide exercises of an appropriate level of difficulty, to provide feedback, and to dispense information. The computer programming the child. In the **LOGO** environment the relationship is reversed: The child is in control: The child programs the computer. And in teaching the computer how to think, children embark on an exploration about how they themselves think. Thinking about thinking turns the child into an epistemologist, an experience not even shared by most adults.

After five years of study with Jean Piaget in Geneva, I came away impressed by his way of looking at children as the active builders of their own intellectual structures. To say that intellectual structures are built by the learner rather than taught by a teacher does not mean that they are built from nothing. Like other builders, children appropriate to their own use materials they find about them, most saliently the models and metaphors suggested by the surrounding culture.

Piaget writes about the order in which the child develops different intellectual abilities. I give more weight than he does to the influence of the materials a certain culture provides in determining that order. For example, our culture is very rich in materials useful for the child's construction of certain components of numerical and logical thinking. Children learn to count; they learn that the result of counting is independent of order and special arrangement; they extend this "conservation" to thinking about the properties of liquids as they are poured and of solids which change their shape. Children develop these components of thinking preconsciously and "spontaneously," that is to say without deliberate teaching. Other components of knowledge, such as the skills involved in doing permutations and combinations, develop more slowly, or do not develop at all without formal schooling.

> ***Many children are held back in their learning because they have a model of learning in which you have either "got it" or "got it wrong." But when you learn to program a computer you almost never get it right the first time.***

The computer presence might have more fundamental effects than did other new technologies, including television and even printing. The metaphor of computer as a mathematics speaking entity puts the learner in a qualitatively new kind of relationship to an important domain of knowledge. Even the best of educational television is limited to offering quantitative improvements in the kinds of learning that existed without it. **Sesame Street** might offer better and more engaging explanations than a child can get from some parents or nursery school teachers, but the child is still in the business of listening to explanations. By contrast, when a child learns to program, the process of learning is transformed. It becomes more active and self directed. The knowledge is acquired for

Copyright 1980, Basic Books, Inc. Used by Permission.

Adapted from Chapter 1 of MINDSTORMS by Seymour Papert. Copyright © 1980 by Basic Books, Inc. By permission of Basic Books, Inc., Publishers, New York.

a recognizable personal purpose. The child does something with it. The new knowledge is a source of power and is experienced as such from the moment it begins to form in the child's mind.

I have spoken of mathematics being learned in a new way. But much more is affected than mathematics. Piaget distinguishes between "concrete" thinking and "formal" thinking. Concrete thinking is already well on its way by the time the child enters first grade at age 6 and is consolidated in the following several years. Formal thinking does not develop until the child is almost 12, give or take a year or two, and some researchers have even suggested that many people never achieve fully formal thinking. I do not fully accept Piaget's distinction, but I am sure that it is close enough to reality to help us make sense of the idea that the consequences for intellectual development of one innovation could be qualitatively greater than the cumulative quantitative effects of a thousand others. My conjecture is that the computer can concretize (and personalize) the formal. Seen in this light, it is not just another powerful educational tool. It is unique in providing us with the means for addressing what Piaget and many others see as the obstacle which is overcome in the passage from child to adult thinking. I believe that it can allow us to shift the boundary separating concrete and formal. Knowledge that was accessible only through formal processes can now be approached concretely. And the real magic comes from the fact that this knowledge includes those elements one needs to become a formal thinker.

This description of the role of the computer is rather abstract. I shall concretize it by looking at the effect of working with computers on two kinds of thinking Piaget associates with the formal stage of intellectual development: combinatorial thinking, where one has to reason in terms of the set of all possible states of a system, and self referential thinking about thinking itself.

In a typical experiment in combinatorial thinking, children are asked to form all the possible combinations (or "families") of beads of assorted colors. It really is quite remarkable that most children are unable to do this systematically and accurately until they are in the fifth or sixth grades. Why should this be? Why does the task seem to be so much more difficult than the intellectual feats accomplished by seven and eight year old children? Is its logical structure essentially more complex? Can it possibly require a neurological mechanism that does not mature until the approach of puberty? I think that a more likely explanation is provided by looking at the nature of the culture. The task of making the families of beads can be looked at as constructing and executing a program, a very common sort of program, in which two loops are nested: Fix a first color and run through all possible second colors, then repeat until all possible first colors have been run through. For someone who is thoroughly used to computers and programming there is nothing "formal" or abstract about this task. For a child in a computer culture it would be as concrete as matching up knives and forks at the dinner table. Even the common "bug" of including some families twice (for example, red-blue and blue-red) would be well known. Our culture is rich in pairs, couples, and one to one correspondences of all sorts, and it is rich in language for talking about such things. This richness provides both the incentive and a supply of models and tools for children to build ways to think about such issues as whether three large pieces of candy are more or less than four smaller pieces. For such problems our children acquire an excellent intuitive sense of quantity. But our culture is relatively poor in models of systematic procedures. Until recently there was not even a name in popular language for programming, let alone for the ideas needed to do so successfully. There is no word for "nested loops" and no word for the double counting bug. Indeed, there are no words for the powerful ideas computerists refer to as "bug" and "debugging."

Our culture is relatively poor in models of systematic procedures.

Without the incentive or the materials to build powerful, concrete ways to think about problems involving systematicity, children are forced to approach such problems, in a groping, abstract fashion. Thus cultural factors can explain the difference in age at which children build their intuitive knowledge of quantity and of systematicity.

While still working in Geneva I had become sensitive to the way in which materials from the then very young computer cultures were allowing psychologists to develop new ways to think about thinking. In fact, my entry into the world of computers was motivated largely by the idea that children could also benefit, perhaps even more than the psychologist, from the way in which computer models seemed to be able to give concrete form to areas of knowledge that had previously appeared so intangible and abstract.

I began to see how children who had learned to program computers could use very concrete computer models to think about thinking and to learn about learning and in doing so, enhance their powers as psychologists and as epistemologists. For example, many children are held back in their learning because they have a model of learning in which you have either "got it" or "got it wrong." But when you learn to program a computer you almost never get it right the first time. Learning to be a master programmer is learning to become highly skilled at isolating and correcting "bugs," the parts that keep the program from working. The question to ask about the program is not whether it is right or wrong, but if it is fixable. If this way of looking at intellectual products were generalized to how the larger culture thinks about knowledge and its acquisition, we might all be less intimidated by our fears of "being wrong." This potential influence of the computer on changing our notion of a black and white version of our successes and failures is an example of using the computer as an "object to think with." It is obviously not necessary to work with computers in order to acquire good strategies for learning. Surely "debugging" strategies were developed by successful learners long before computers existed. But thinking about learning by analogy with developing a program is a powerful and accessible way to get started on becoming more articulate about one's debugging strategies and more deliberate about improving them.

My discussion of a computer culture and its impact on thinking presupposes a massive penetration of powerful computers into people's lives. That this will happen there can be no doubt. The calculator, the electronic game, and the digital watch were brought to us by a technical revolution that rapidly lowered prices for electronics in a period when all others were rising with inflation. That same technological revolution, brought about by the integrated circuit, is now bringing us the personal computer.

There really is no disagreement among experts that the cost of computers will fall to a level where they will enter everyday life in vast numbers. Some will be there as computers proper, that is to say, programmable machines. Others might appear as games of ever increasing complexity and in automated supermarkets where the shelves, maybe even the cans, will talk. There is no doubt that the material surface of life will become very different for everyone, perhaps most of all for children. But there has been a significant difference of opinion about the effects this computer presence will produce. I would distinguish my thinking from two trends of thinking which I refer to here as the "skeptical" and the "critical."

Skeptics do not expect the computer presence to make much difference in how people learn and think. I have formulated a number of possible explanations for why they think as they do. In some cases I think the skeptics might conceive of education and the effect of computers on it too narrowly. Instead of considering general cultural effects, they focus attention on the use of the computer as a device for programmed instruction. Skeptics then conclude that while the computer might produce some improvements in school learning, it is not likely to lead to fundamental change. In a sense, too, I think the skeptical view derives from a failure to appreciate just how much Piagetian learning takes place as a child grows up. If a person conceives of children's intellectual development (or, for that matter, moral or social development) as deriving chiefly from deliberate teaching, then such a person would be likely to underestimate the potential effect that a massive presence of computers and other interactive objects might have on children.

Knowledge that was accessible only through formal process can now be approached concretely.

The critics, on the other hand, do think that the computer presence will make a difference and are apprehensive. For example, they fear that more communication via computers might lead to less human association and result in social fragmentation. As knowing how to use a computer becomes increasingly necessary to effective social and economic participation, the position of the underprivileged could worsen, and the computer could exacerbate existing class distinctions. As to the political effect computers will have, the critics' concerns resonate with Orwellian images of a 1984 where home computers will form part of a complex system of surveillance and thought control. Critics also draw attention to potential mental health hazards of computer penetration. Some of these hazards are magnified forms of problems already worrying many observers of contemporary life; others are problems of an essentially new kind. A typical example of the former kind is that our grave ignorance of the psychological impact of television becomes even more serious when we contemplate an epoch of super TV. The holding power and psychological impact of the television show could be increased by varying the content to suit the tastes of each individual viewer, and by the show becoming interactive, drawing the viewer into the action. Critics already cite cases of students spending sleepless nights riveted to the computer terminal, coming to neglect both studies and social contact.

In the category of new problems, critics have pointed to the influence of the allegedly mechanized thought processes on how people think. Marshall MacLuhan's dictum that "the medium is the message" might apply here: If the medium is an interactive system that takes in words and speaks back like a person, it is easy to get the message that machines are like people and that people are like machines. What this might do to the development of values and self image in growing children is hard to assess. But it is not hard to see reasons for worry.

Despite these concerns I am essentially optimistic—some might say utopian—about the effects of computers on society. I do not dismiss the arguments of the critics. On the contrary, I too see the computer presence as a potent influence on the human mind. I am very much aware of the holding power of an interactive computer and of how taking the computer as a model can influence the way we think about ourselves. In fact the work on **LOGO** to which I have devoted much of the past ten years consists precisely of developing such forces in positive directions. For example, the critic is horrified at the thought of a child hypnotically held by a futuristic, computerized super pinball machine. In the **LOGO** work we have invented versions of such machines in which powerful ideas from physics or mathematics or linguistics are imbedded in such a way that permits the player to learn them in a natural fashion, analogous to how a child learns to speak. The computer's "holding power," so feared by critics, becomes a useful educational tool. Or take another, more profound example. The critic is afraid that children will adopt the computer as model and eventually come to "think mechanically" themselves. Following the opposite tack, I have invented ways to take educational advantage of the opportunities to master the art of *deliberately* thinking like a computer, according, for example, to the stereotype of a computer program that proceeds in a step-by-step, literal, mechanical fashion. There are situations where this style of thinking is appropriate and useful. Some children's difficulties in learning formal subjects such as grammar or mathematics derive from their inability to see the point of such a style.

A second educational advantage is indirect but ultimately more important. By deliberately learning to imitate mechanical thinking, the learner becomes able to articulate what mechanical thinking is and what it is not. The exercise can lead to greater confidence about the ability to choose a cognitive style that suits the problem. Analysis of "mechanical thinking" and how it is different from other kinds and practice with problem analysis can result in a new degree of intellectual sophistication. By providing a very concrete, down to earth model of a particular style of thinking work with the computer can make it easier to understand that there is such a thing as a "style of thinking." And giving children the opportunity to choose one style or another provides an opportunity to develop the skill necessary to choose between styles. Thus instead of inducing mechanical thinking, contact with computers could turn out to be the best conceivable antidote to it. And for me what is most important in this is that through these experiences these children would be serving their apprenticeship as epistemologists, that is to say learning to think articulately about thinking.

The intellectual environments offered to children by today's cultures are poor in opportunities to bring their thinking about thinking into the open, to learn to talk about it and test their ideas by externalizing them. Access to computers can dramatically change this situation. Even the simplest Turtle work can open new opportunities for sharpening one's thinking about thinking: Programming the Turtle begins by making one reflect on how one does oneself what one would like the Turtle to do. Thus teaching the Turtle to act or to "think" can lead one to reflect on one's own actions and thinking. And as children move on, they program the computer to make more complex decisions and find themselves engaged in reflecting on more complex aspects of their own thinking.

In teaching the computer how to think, children embark on an exploration about how they themselves think.

In short, while the critic and I share the belief that working with computers can have a powerful influence on how people think, I have turned my attention to exploring how this influence could be turned in positive directions.

The central open questions about the effect of computers on children in the 1980s are these: Which people will be attracted to the world of computers, what talents will they bring, and what tastes and ideologies will they impose on the growing computer culture? I have observed children in **LOGO** environments engaged in self-referential discussions about their own thinking. This could happen because the **LOGO** language and the Turtle were designed by people who enjoy such discussion and worked hard to design a medium that would encourage it. Other designers of computer systems have different tastes and different ideas about what kinds of activities are suitable for children. Which design will prevail, and in what subculture, will not be decided by a simple bureaucratic decision made, for example, in a government Department of Education or by a committee of experts. Trends in computer style will emerge from a complex web of decisions by foundations with resources to support one or another design, by corporations who may see a market, by schools, by individuals who decide to make their career in the new field of activity, and by children who will have their own say in what they pick up and what they make of it. People often ask whether in the future children will program computers or become absorbed in pre-programmed activities. The answer must be that some children will do the one, some the other, some both, and some neither. But which children, and most importantly, which social classes of children, will fall into each category will be influenced by the kind of computer activities and the kind of environments created around them.

As an example, we consider an activity which may not occur to most people when they think of computers and children: the use of the computer as a writing instrument. For me, writing means making a rough draft and refining it over a considerable period of time. My image of myself as a writer includes the expectation of an "unacceptable" first draft that will develop with successive editing into presentable form. But I would not be able to afford this image if I were a third grader. The physical act of writing would be slow and laborious. I would have no secretary. For most children rewriting a text is so laborious that the first draft is the final copy, and the skill of rereading with a critical eye is never developed. This changes dramatically when children have access to computers capable of manipulating text. The first draft is composed at the keyboard. Corrections are made easily. The current copy is always neat and tidy. I have seen children move from total rejection of writing to an intense involvement (accompanied by rapid improvement of quality) within a few weeks of beginning to write with a computer. Even more dramatic changes are seen when the child has physical handicaps that make writing by hand more than usually difficult or even impossible.

This use of computers is rapidly being adopted wherever adults write for a living. Most newspapers now provide their staff with "word processing" computer systems. Many writers who work at home are acquiring their own computers, and the computer terminal is steadily displacing the typewriter as the secretary's basic tool. The image of children using the computer as a writing instrument is a particularly good example of my thesis that what is good for professionals is good for children. But this image of how the computer might contribute to children's mastery of language is dramatically opposed to the one that is taking root in most elementary schools. There the computer is seen as a teaching instrument. It gives children practice in distinguishing between verbs and nouns, in spelling, and in answering multiple choice questions about the meaning of pieces of text. As I see it, this difference is not a matter of a small and technical choice between two teaching strategies. It reflects a fundamental difference in educational philosophies. More to the point, it reflects a difference in view on the nature of childhood. I believe that the computer as writing instrument offers children an opportunity to become more like adults, indeed like advanced professionals, in their relationship to their intellectual products and to themselves. In doing so, it comes into head-on collision with the many aspects of school whose effect, if not whose intention, is to "infantilize" the child.

Word processors *can* make a child's experience of writing more like that of a real writer. But this can be undermined if the adults surrounding the child fail to appreciate what it is like to be a writer. For example, it is only too easy to imagine adults, including teachers, expressing the view that editing and re-editing a text is a waste of time ("Why don't you get on to something new?" or "You aren't making it any better, why don't you fix your spelling?").

The critic is horrified at the thought of a child hypnotically held by a futuristic, computerized super pinball machine.

As with writing, so with music making, games of skill, complex graphics, whatever: The computer is not a culture unto itself but it can serve to advance very different cultural and philosophical outlooks. For example, one could think of the Turtle as a device to teach elements of the traditional curriculum, such as notions of angle, shape, and coordinate systems. And in fact, most teachers who consult me about its use are trying to use it in this way. Of course the Turtle can help in the teaching of traditional curriculum, but I have thought of it as a vehicle for Piagetian learning, which to me is learning without curriculum.

There are those who think about creating a "Piagetian curriculum" or "Piagetian teaching methods." But to my mind these phrases and the activities they represent are contradictions in terms. I see Piaget as the theorist of learning without curriculum and the theorist of the kind of learning that happens without deliberate teaching. To turn him into the theorist of a new curriculum is to stand him on his head.

But "teaching without curriculum" does not mean spontaneous, free form classrooms or simply "leaving the child alone." It means supporting children as they build their own intellectual structures with materials drawn from the surrounding culture. In this model, educational intervention means changing the culture, planting new constructive elements in it and eliminating noxious ones. This is a more ambitious undertaking than introducing a curriculum change, but one which is feasible under conditions now emerging.

Suppose that thirty years ago an educator had decided that the way to solve the problem of mathematics education was to arrange for a significant fraction of the population to become fluent in (and enthusiastic about) a new mathematical language. The idea might have been good in principle, but in practice it would have been absurd. No one had the power to implement it. Now things are different. Many millions of people are learning programming languages for reasons that have nothing to do with the education of children. Therefore, it becomes a practical proposition to influence the form of the languages they learn and the likelihood that their children will pick up these languages.

Throughout the course of this chapter I have been talking about the ways in which choices made by educators, foundations, governments, and private individuals can affect the potentially revolutionary changes in how children learn. But making good choices is not always easy, in part because past choices can often haunt us. There is a tendency for the first usable, but still primitive, product of a new technology to dig itself in. I have called this phenomenon the QWERTY phenomenon.

The top row of alphabetic keys of the standard typewriter reads QWERTY. For me this symbolizes the way in which technology can all too often serve not as a force for progress but for keeping things stuck. The QWERTY arrangement has no rational explanation, only a historical one. It was introduced in response to a problem in the early days of the typewriter: The keys used to jam. The idea was to minimize the collision problem by separating those keys that followed one another frequently. Just a few years later, general improvements in the technology removed the jamming problem, but QWERTY stuck. Once adopted, it resulted in many millions of typewriters and a method (indeed a full blown curriculum) for learning typing. The social cost of change (For example, putting the most used keys *together* on the keyboard) mounted with the vested interest created by the fact that so many fingers now knew how to follow the QWERTY keyboard. QWERTY has stayed on despite the existence of other, more "rational" systems. On the other hand, if you talk to people about the QWERTY arrangement they will justify it by "objective" criteria. They will tell you that it "optimizes this" or it "minimize that." Although these justifications have no rational foundation, they illustrate a process, a social process, of myth construction that allows us to build a justification for primitivity into any system. I think we are well on the road to doing exactly the same thing with the computer. We are in the process of digging ourselves into an anachronism by preserving practices that have no rational basis beyond their historical roots in an earlier period of technological and theoretical development.

The use of computers for drill and practice is only one example of the QWERTY phenomenon in the computer domain. Another example occurs even when attempts are made to allow students to learn to program the computer. Learning to program a computer involves learning a "programming language." There are many such languages—for example, Fortran, Pascal, Basic, Smalltalk, and Lisp, and the lesser known language **LOGO**, which our group has used in most of our experiments with computers and children. A powerful QWERTY phenomenon is to be expected when we choose the language in which children are to learn to program computers. I shall argue in detail that the issue is consequential. A programming language is like a natural, human language in that it favors certain metaphors, images, and ways of thinking. It would seem to follow that educators interested in using computers and sensitive to cultural influences would pay particular attention to the choice of language. But nothing of the sort has happened. On the contrary, educators, too timid in technological matters or too ignorant to attempt to influence the languages offered by computer manufacturers, have accepted certain programming languages in much the same way as they accepted the QWERTY keyboard. An informative example is the way in which the programming language Basic has established itself as the obvious language to use in teaching American children how to program computers. The relevant technical information is this: A very small computer can be made to understand Basic, while other languages demand more from the computer. Thus, in the early days when computer power was extremely expensive, there was a genuine technical reason for the use of Basic, particularly in schools where budgets were always tight. Today, and in fact for several years now, the cost of computer memory has fallen to the point where any remaining economic advantages of using Basic are insignificant. Yet in most high schools, the language remains almost synonymous with programming, despite the existence of other computing languages that are demonstrably easier to learn and are richer in the intellectual benefits that can come from learning them. The situation is paradoxical. The computer revolution has scarcely begun, but it is already breeding its own conservatism. Looking more closely at Basic provides a window on how a conservative social system appropriates and tries to neutralize a potentially revolutionary instrument.

Giving children the opportunity to choose one style or another provides an opportunity to develop the skill necessary to choose between styles.

Basic is to computation what QWERTY is to typing. Many teachers have learned Basic, many books have been written about it, many computers have been built in such a way that Basic is "hardwired" into them. In the case of the typewriter, we noted how people invent "rationalizations to justify the status quo. In the case of Basic, the phenomenon has gone much further, to the point where it resembles ideology formation. Complex arguments are invented to justify features of Basic that were originally included because the primitive technology demanded them or because alternatives were not well enough known at the time the language was designed.

An example of Basic ideology is the argument that Basic is easy to learn because it has a very small vocabulary. Its small vocabulary can be learned quickly enough. But using it is a different matter. Programs in Basic acquire so labyrinthine a structure that only the most motivated and brilliant ("mathematical") children do learn to use it for more than trivial ends.

One might ask why the teachers do not notice the difficulty children have in learning Basic. The answer is simple: Most teachers do not expect high performance from most students, especially in a domain of work that appears to be as "mathematical" and "formal" as programming. Thus the culture's general perception of mathematics as inaccessible bolsters the maintenance of Basic, which in turn confirms these perceptions. Moreover, the teachers are not the only people whose assumptions and prejudices feed into the circuit that perpetuates Basic. There are also the computerists, the people in the computer world who make decisions about which languages their computers will speak. These people, generally engineers, find Basic quite easy to learn, partly because they are accustmed to learning such very technical systems and partly because Basic's sort of simplicity appeals to their system of values. Thus, a particular subculture, one dominated by computer engineers, is influencing the world of education to favor those school students who are most like that subculture. The process is tacit, unintentional: It has never been publically articulated, let alone evaluated. In all of these ways, the social embedding of Basic has far more serious consequences than the "digging in" of QWERTY.

There are many other ways in which the attributes of the subcultures involved with computers are being projected onto the world of education. For example, the idea of the computer as an instrument for drill and practice that appeals to teachers because it resembles traditional teaching methods also appeals to the engineers who design computer systems: Drill and practice applications are predictable, simple to describe, efficient in use of the machine's resources. So the best engineering talent goes into the development of computer systems that are biased to favor this kind of application. The bias operates subtly. The machine designers do not actually decide what will be done in the classrooms. That is done by teachers and occasionally even by carefully controlled research experiments. But there is an irony in these controlled experiments. They are very good at telling whether the small effects seen in best scores are real or due to chance. But they have no way to measure the undoubtedly real (and probably more massive) biases built into the machines.

We have already noted that the conservative bias being built into the use of computers in education has also been built into other new technologies. The first use of the new technology is quite naturally to do in a slightly different way what had been done before without it. It took years before designers of automobiles accepted the idea that they were cars, not "horseless carriages," and the precursors of modern motion pictures were plays acted as if before a live audience but actually in front of a camera. A whole generation was needed for the new art of motion pictures to emerge as something quite different from a linear mix of theater plus photography. Most of what has been done up to now in the name of "educational technology" or "computers in education" is still at the stage of the linear mix of old instructional methods with new technologies.

We are at a point in the history of education when radical change is possible, and the possibility for that change is directly tied to the impact of the computer. Today what is offered in the educational "market" is largely determined by what is acceptable to a sluggish and conservative system. But this is where the computer presence is in the process of creating an environment for change. Consider the conditions under which a new educational idea can be put into practice today and in the near future. Let us suppose that today I have an idea about how children could learn mathematics more effectively and more humanely. And let us suppose that I have been able to persuade a million people that the idea is a good one. For many products such a potential market would guarantee success. Yet in the world of education today this would have little clout: A million people across the nation would still mean a minority in every town's school system, so there might be no effective channel for the million voices to be expressed. Thus, not only do good educational ideas sit on the shelves, but the process of invention is itself stymied. This inhibition of invention in turn influences the selection of people who get involved in education. Very few with the imagination, creativity, and drive to make great new inventions enter the field. Most of those who do are soon driven out in frustration. Conservatism in the world of education has become a self-perpetuating *social* phenomenon.

Basic is to computation what QWERTY is to typing.

Fortunately, there is a weak link in the vicious circle. Increasingly, the computers of the near future will be the private property of individuals, and this will gradually return to the individual the power to determine patterns of education. Education will become more of a private act, and people with good ideas, different ideas, exciting ideas will no longer be faced with a dilemma where they either have to "sell" their ideas to a conservative bureaucracy or shelve them. They will be able to offer them in an open marketplace directly to consumers. There will be new opportunities for imagination and originality. There might be a renaissance of thinking about education. ☐

Classroom Computers: Beyond the 3 R's

Fred Bell

The classroom computer should, and can, go far beyond rote computer-aided instruction by teaching the student to analyze, evaluate and develop complex skills. Perhaps, as a result, the long-awaited "revolution in education" will be here sooner than predicted.

Since 1965, computer buffs, myself included, have been promising a revolution in education because computers are going to school. But where is this revolution? Certainly there has been at least a modest learning revolution; this is apparent from the many people who are learning about computers and using them to learn other things. In spite of the fact that some worthwhile applications are being done with computers in a few exemplary schools, this learning revolution has yet to take place in most schools. There has been an evolution in school learning (at least in many schools) that can be attributed, in part, to computer technology, but no real revolution.

Before considering the potential revolutionary effects of personal computers upon education, it is helpful to differentiate between school learning and out-of-school learning. The two are not always the same. We tend to learn things away from school when we want or need to learn them and we do so in our own way and at our own speed. This kind of learning has advantages and disadvantages. One advantage comes from higher motivation which encourages more inspired and efficient learning. On the other hand, the tendency to avoid difficult or uninteresting tasks may result in not learning some very useful and important things. Consequently schools are useful in coercing students, hopefully in a friendly and interesting way, into learning some things that are good for them which may not be learned otherwise. Out-of-school learning can be both good and bad, but so can in-school learning, which gets us to personal-computers and the education revolution.

Personal Computers and Dollars

One of the big reasons why personal computers may catalyze a revolution in our schools is that they are relatively cheap and should get even cheaper. Any family that can afford two color TV sets can now afford one color TV and a personal computer. Of course, any high school that could scrape up $10,000 per year for each of 10 years from a $1,000,000 per year budget could have had nearly all of its students using a minicomputer since 1969. (See James Saunder, **Mathematics Teacher**, May, 1978, pp. 443-447.) Fortunately, a family's decision-making processes in buying a personal computer are less cumbersome than a school's. Unfortunately for school students, as David Lichtman found (**Creative Computing**, January, 1979, page 48), educators are less enthusiastic about the computer's role in society and its potential for improving education than the general public.

But now, with low-cost personal computers, good computer applications may increase in schools. Home-computing enthusiasts have already begun to take learning out of the schools and are putting some of it back into the home where it belongs. Conversely, as more and more personal computers come to school, teachers can bring some of this good "street learning" back into the schools for the benefit of all students. Only $500 remaining in an equipment-and-supplies account at the end of the fiscal year can buy the first of many personal computers for student and teacher use.

History shows that many technological innovations that could be quite useful in promoting learning in schools do not get much use in schools until after they are common in homes and on the streets; for example, TV sets, audio recorders and hand-held calculators. Now that personal computers are "on the streets," we are beginning to see them filtering into schools. But will they be able to revolutionize education in schools? TV sets, audio recorders, calculators, and even minicomputers, while affecting what goes on in schools, failed to revolutionize education. Can we expect the personal computer to become a revolutionary agent? Yes, I think we can.

Personal Computers and Motivation

One of the most serious problems in schools is that of motivating stu-

---

Fred Bell, Professor of Mathematics Education, University of Pittsburgh, Division of Teacher Development, 4A01 Forbes Quadrangle, Pittsburgh, PA 15260.

From *Creative Computing*, Volume 5, Number 9, pages 68–70. Copyright © 1981 by Creative Computing. All rights reserved. Reprinted with permission.

dents; that is, making them want to learn what teachers try to teach. The motivation problem occurs because sometimes teachers want to teach things (for good reasons) to students who do not particularly care to learn at the time. Perhaps the best hope for motivating students to learn in school is to pay attention to the nature of out-of-school learning. It appears that people learn non-survival things away from school for several reasons: (1) People learn to make things work (such as cars and computers) because they like to have control over impressive machines. (2) People learn to build model airplanes, radios, bookcases, etc. because they find satisfaction in creating something from nothing, or next to nothing. (3) People teach classes, give speeches, and write articles because they like to share their opinions and knowledge with others and possibly influence other people's opinions. Many things are learned because people enjoy the recognition and approval of other people. (5) Other activities that are not necessary for survival are carried out for relaxation, enjoyment, and self satisfaction.

But why do so many students dislike learning in school? First, students seldom have control over the academic machinery of schools; that is, the classroom learning environment. Second, creating and building tangible things occurs all to seldom in most classes. Third, students' opinions tend to be overshadowed by teachers' opinions in many classrooms. Fourth, many students get low grades in school, which interferes with their quest for recognition and approval. Fifth, much of what students have to do in school is neither relaxing, enjoyable, nor self-satisfying.

But how can a few personal computers in a classroom solve these motivational problems for students and teachers? Well, computers and computer-enhanced learning are not educational panaceas, but they can give students some real control over what they learn and how they learn it. Making a computer (an electronic monster) do one's bidding is fun for many people, in spite of the fact that it is, at times, tedious and frustrating. Writing a computer program and making it do what it is supposed to do is creating something — both a physical and an intellectual creation. Most people (including teachers and students) are impressed by good interactive computer games, simulations and tutorials, which provide recognition and influence for their creators. Finally, messing around, in a meaningful way of course, with a personal computer can be relaxing and enjoyable, in spite of many minor, temporary frustrations and aggravations.

Therefore, we find that personal computers in the hands of students in school can remove some of the artificial constraints of typical classroom environments and replace them with some of the personal freedoms inherent in many non-school learning situations.

Personal Computers and Learning

What is learned in school? English, reading, writing, arithmetic, French, history, etc.? Yes, these **are** some of the subjects that are taught but students should learn many other things that subsume all subjects. That is, students need to study each subject in a manner that permits them to function at all of the following cognitive levels:

knowledge
understanding
application
analysis
synthesis
evaluation
problem solving
knowing how to learn
creating knowledge

Schools are fairly good at imparting knowledge (i.e., "George Washington was the first U.S. president") and understanding (i.e., "2 + 3 = 5 because 2 marbles together with 3 marbles is 5 marbles"). However, schools are only moderately successful at teaching applications (out-of-school uses for each subject), analysis (breaking a skill or conceptual structure into its parts), synthesis (building complex skills or conceptual structures from simpler things), and evaluation (comparing skills and structures and making judgments about them). Schools and teachers have even less success at teaching students the skills and heuristic procedures of problem solving, how to learn independently of teachers and courses, and ways of conducting the research and explorations that go into creating knowledge.

During the past 15 years we have demonstrated, through many dramatic examples, that computers can be used in schools to help teach knowledge, understanding, and applications of various subjects—things that were being done fairly well without computers. This is the evolutionary aspect of computers in education. But what about the higher-level cognitive activities, those things that we haven't been able to teach very successfully in school? Herein lies the true power of computers (especially personal computers) to really revolutionize learning and teaching in schools.

Writing a computer program requires analysis and synthesis of the subject under consideration as well as the program itself. A student cannot write a program to tutor others, play a game, simulate a situation, or solve an exercise without analyzing the topic being studied and synthesizing it into a coherent teaching/learning program. The synthesis required in writing the program properly and the analysis in debugging it provides additional practice at synthesizing and analyzing. Since many non-tutorial computer programs are higher-lever applications of topics, the student programmer must evaluate the approriateness of alternative approaches to the topic and the program. When a student writes computer programs to extend and clarify topics in school, the six steps in problem-solving (posing the problem, precisely defining the problem, gathering information, developing a solution strategy, finding the solution and checking the solution) must be carried out. On the other hand, most so-called "problems" in textbooks are really exercises for practicing skills, which require only one of the six steps of problem solving; namely, finding the answer. After several years of working with people in Project Solo at the University of Pittsburgh, we found that many students and teachers could carry out independent research of their own choosing in computer-enhanced learning environments. That is, these people were creating knowledge and learning how to learn independent of people who were labeled as the teachers and rooms that were called classrooms.

Now personal computers can bring the Solo concept of high-level, self-motivated learning out of the research-and-development laboratory and put it in the hands of large numbers of students and teachers in school classrooms.

Carrying Out the Revolution

Even before the advent of personal computers (as early as 1972), the computer technology and courseware existed for a revolution in teaching and learning in schools. Now personal computers with their low costs, easy accessibility, total dedication to the user, and person-on-the-street popularity may provide the long-awaited catalyst that is needed to make some dramatic changes in how computers are used in schools. In a few years large numbers of students entering high school will be as familiar with a computer as they are now with a TV set, probably more so since they will have actively programmed a computer, in comparison to watching television passively.

As a consequence of the popularity of television, Americans are accused of having become spectators rather than participants in life. Personal computing certainly requires active intellectual participation on the part of the user. I have yet to hear of anyone dozing off while sitting in front of a personal computer.

For several years mathematics teachers worried about whether kids should be allowed to use hand-held calculators in school. The popularity of calculators outside school quickly settled that issue. Nearly every family had a calculator. Pre-school children played with them and students brought them to school. Teachers could not ignore calculators because it was impossible to keep them out of school; so now they are trying to determine how best to incorporate calculators and calculator-related skills into the school mathematics curriculum. Even if people try to keep personal computers out of schools, they are going to fail. In a few years, when they are more efficiently packaged and even less expensive, personal computers can fill the "lunchbox-technology" void created by school-lunch programs. Instead of a lunchbox, students will be carrying a PET or TRS-80 computer on a handle to school. When this time comes, an Apple for the teacher will really help a kid get a better grade in school. ☐

References

Barstow, Daniel. "Computers and Education: Some Questions of Values," **Creative Computing** 5: 116-119; February 1979.

Bell, Frederick H. "Why Is Computer-Related Learning So Successful", **Educational Technology** 14:15-18; December 1974.

Bell, Frederick H. "Can Computers Really Improve School Mathematics?", **The Mathematics Teacher** 71:428-33; May 1978.

Dwyer, Thomas A. "Soloworks: Computer-Based Laboratories for High School Mathematics," **School Science and Mathematics** 75:93; January 1975.

Lichtman, David. "Survey of Educator's Attitudes Toward Computers," **Creative Computing** 5:46; January 1979.

Molnar, Andrew R. "The Next Great Crisis in American Education: Computer Literacy," **THE Journal** 5:35-38; July/August 1978.

ABZ COMPUTER CONVEYOR SYS.

"Mr. Axilbroad will see you now, sir. Keep your seat."

©Creative Computing

2 Perspectives of Gifted Children and Youth

Two of the three articles in this section were written especially for this book. The two high school students who were invited to contribute were selected because their experiences with computers were known to have had a significant impact on their lives. Although their experiences differ in some respects—one has his own personal computer at home, while the other has access to large main frame computers—both know that their future lies in the field of computer technology.

One of the authors, Stephen Bloch, a high school senior, recognizes the same qualities in computer experiences as do Papert and Bell. Bloch comments.

> Teaching the skills, habits, and disciplines of computer programming improves a child's (or an adult's) ability to think, and these skills should be taught for the same reasons the so-called Great Books should.
>
> In computer programming, nothing can be glossed over, nothing can be left to cure itself, and everything must be accounted for.
>
> The invention of computers has caused irrevocable changes in human society, and within 40 years will bring about changes we cannot conceive of today. Even if it were desirable to eliminate them from our lives, we cannot reject them now without crippling civilization for decades. Obviously, therefore, we must adapt to them: learn why they work, learn how to use them, and take advantage of their growth by learning from them. Computers are not "newfangled" things that will go out of style in 10 years. If we do not teach our kids to adapt to the rapid changes the next 40 years will bring, we are making them maladjusted to their world from the start.

For Christopher Karnes, an 11th grade student, his computer has been like a magic carpet. As a result of his interest, he has participated in activities he probably never would have engaged in otherwise. For example, Chris experienced the challenge of taking college courses and the pleasure of successful completion while still a high school sophomore. He designed a research study using his computer which earned him a first place trophy in the State Science Fair. Chris believes computer experiences can be used successfully by all children and youth because computer programming is a fascinating, creative, and intellectual challenge.

The final article in this section describes the computer experiences of Safi Bahcall, an 11-year-old eighth grade student. The article also presents two programs Safi has developed. His introduction to the world of computers is similar to that of Karnes in that they both have TRS-80 personal computers. Safi's programs were included to illustrate what level of accomplishment is possible when children are turned loose with a computer.

Reflections of A Computer Language Nut

STEPHEN BLOCH

My introduction to computers came in December 1979. There was a computer room in my high school, and through its windows I had seen some friends of mine typing away feverishly at what I recognized after some thought as computer terminals. It took me about a week to find enough nerve to knock at the door. Interestingly enough, it was not the computer itself that I was afraid of. I had read enough about how computers were a man-made invention completely controlled by humans to be interested, not frightened, by the prospect of controlling them myself. I was afraid of the rites of passage into the "computer clique," which I envisioned as some sort of Pythagorean mystical society. Fingers crossed behind my back, sweat dripping from my brow, I raised a fist to knock at the door and was greeted with the anticlimactic news that I would have to talk to a math teacher down the hall to obtain an ID enabling me to use the computer. Oh Brave New World!

The computer I was first introduced to was a "mini" called the Hewlett-Packard 2000. It could communicate with up to 32 users at a time, most of whom were connected by telephone cable from all over the county. This particular computer "understood" only one of the many artificial languages on the market today, an educational language developed in 1964 called BASIC. BASIC is more readable and hence easier for a beginner to understand than FORTRAN, LisP, or machine language. However, its unstructured style is responsible for most of the sloppy programming in the world today, because so many programmers like myself first learned BASIC. Its greatest selling point is that it is by far the most widely known language for personal and small business computers in the world. Regardless of the language, however, the multi-user system was appropriate for a beginner because there were always plenty of experienced people to talk to via a message-sending program built into the system. The ability to call on more experienced people for help is important in learning to program and use a computer, and every multi-user system I have worked on has had an easily accessible program to perform this message-sending function.

Exploring Computer Languages

During the first year of my computer use I was restricted to BASIC, though I did use two of the dozens of different versions of it. In 11th grade I took the Computer Science II course and was introduced to FORTRAN, one of the oldest computer languages, dating to the late 1950's. I didn't like it, partially because the nearest computer that ran it was a different model that had more complicated commands and no documentation to explain how to use them. Because of this association, and the fact that it contained few features our version of BASIC lacked, I still dislike FORTRAN. However, the idea of the existence of languages other than BASIC, to which I had grown so accustomed that I had almost forgotten it was an invented language to begin with, intrigued me.

I started trying to write my own language, modestly called BLOCH (Branch-Loop Oriented Computer Handler) and found that I could not write it efficiently without a language specifically designed for writing computer languages. Therefore, I started learning SPL, Systems Programming Language. I had little success writing in it, but its modular structure was quite different and far more logical than the linear structure of BASIC and FORTRAN. Then I tried PASCAL, a similar modular instruction language my father told me about. I visited my cousin at MIT and found that his computer science course used one of the oldest structured languages, LisP (List Processing language). I started trying to learn that, which was difficult because I had no LisP instruction manual, and of the two in existence, one was obsolete and the other incomplete.

I joined an Explorer post in my area which met every week at the Honeywell building for 2 hours to learn about computers, and found to my delight that their computer ran LisP. A friend in my Computer Science class mentioned a languaged called FORTH, in which programs consisted entirely of words defined in terms of each other, and this interested me. All of these languages, collectively referred to as "high-level languages," were based upon and defined in a machine language, which is fast, compact, and utterly unreadable. Then I started to wonder what such a low-level language was really like. In short, I went language crazy, and I still am to some extent.

When introduced to computers, most people find some area of special interest in which they, too, "go crazy," though not many are language nuts as I am. For example, one of my friends does only what programming is required for class and spends most of his time improving and testing a homemade microcomputer, which he has now built up to the level of a relatively cheap hand-held programmable calculator, but which has far more capacity for expansion. My Explorer post is divided into three groups, roughly equivalent to patrols in Boy or Girl Scouts, which concern themselves respectively with software, hardware, and simulations (programming a computer to act like a different type of computer or to understand a different language) and although every Explorer does some programming, we work mostly within our groups.

One effect seen in people, especially teenagers, who spend a lot of time with computers is somewhat frightening: In some cases they forget how to communicate with everyday, illogical humans. The closest approach to this I have seen myself is a tendency to semiconsciously speak in jargonese and buzzwords ("trouble utilizing the FOPEN intrinsic in an SPL sysdump program that breaks security by FCOPYing the low-memory end of the working-directory RAM") with people who know little or nothing about computers. This effect can be avoided if the person concerned works on computers less than about four hours a day, but its existence gives one pause.

The Attraction of Computers

We have seen that young people are often naturally attracted to computers for a variety of very basic reasons ("Ooh, look at the pretty colors!" "I take the center square..." "We'll show those Klingons! Ahead Warp Factor Four!" "If I bishop to King's knight five, he'll probably ..." "Where in the user-input routine do I want the error-trap subroutine to return to?"). Computers are virtually impossible to destroy or even mildly confuse without a long-term, conscious effort, and even with one—I've tried! The most important requirement to learning about or even with computers is freeing oneself from hangups about robot revolutions and looking foolish, so that one can concentrate on learning from and teaching these idiot-savant machines. Now that you know more or less how to learn about computers, it might be nice to know why as well.

The Need for Computer Literacy

Over the past few years numerous reasons to glorify computers have been found, so I'll skip briefly through the more obvious ones to one I feel is both more valid and more important for gifted children than all the rest combined. The use of the computer as a teacher's aide depends on optical-scanner forms, from general ones a teacher can use for multiple-choice tests to nationally standardized tests like the SRA/ITED, the PSAT/NMSQT, and the SAT. Without black No. 2 pencils to fill in the entire circle but not lap outside it (see example on your test booklet, back page), our colleges and high schools would come to a screeching halt within a month or two!

Many high schools and most colleges have kept grading, scheduling, and attendance information on computer files for years. But a system in which teachers use computers regularly and students not at all will clearly give young people the impression that computers are "grown-up things" which they are not allowed to use. Admittedly, the feeling of forbiddenness often adds a thrill of danger, but in this case it also prevents students from asking their teachers for help.

One commonly cited reason for teaching kids about computers is the immense practical value such skills will have in a computer-oriented world, both in job-finding and in everyday life. However, a major goal of today's programmers is to make computer programs more and more "friendly" so computer training beyond the point of losing the fear of computers will have little practical everyday use.

A more significant development, however, is the series of programs written in the last 5 years, as program generators. A person with little or no skill at programming can specify in everyday English what he or she wants a program to do, and a program generator such as The Last One will write it quickly. The Last One works best for applications programs, those which serve a specific purpose such as playing a game or predicting the position of a spaceship, but YACC (Yet Another Compiler Constructor) or something similar will quickly write an interpreting program for almost any language you can specify. What all this means, according to many computer authorities, is that in 20 to 30 years there will be little need for human programmers at all.

The purpose of bringing up children with computers may not, therefore, be to prepare them for the abundant jobs of a computer-oriented world; such a world would indeed have plentiful jobs in computer operation and programming. A technological advance cannot be considered as having been truly accepted until it becomes ludicrous to even talk about being "oriented" toward it. One might call our society "television oriented" or even "automobile oriented," but only the most precise anthropologists would call it "wheel oriented." Granted, wheels have been around for thousands of years, but in view of the incredibly accelerating pace of technology in this century, I doubt that computers will take over 50 years to reach the same level of acceptance and ignorability.

As far as I can see, the greatest boon of computer literacy is a fairly intangible one. Teaching the skills, habits, and disciplines of computer programming improves a child's (or an adult's) ability to think, and these skills should be taught for the same reasons the so-called Great Books should. For example, Euclid's *Elements* contains little factual knowledge that has not been rehashed and then taught in an easier fashion, not to mention its relative inapplicability to everyday life, dealing as it does with a system that does not represent reality, as Euclid knew quite well. The benefit of the *Elements* is simply that it teaches the habits of clear, rigorous thought in the same way computer programming does.

In computer programming, nothing can be glossed over, nothing can be left to cure itself, and everything must be accounted for. If seemingly insignificant problems are ignored, they often manifest themselves in grandiose ways, commonly known as bugs or zarkas. (A simple example might be a chess program that, whenever you legally take one of its pieces, eliminates instead *your* corresponding piece.) As a result, programmers, even if they communicate only with "friendly" computers in spoken English, gradually learn to think more clearly and with greater comprehension of the abstract effects of their actions.

Children and Computers

If computer use and familiarity are indeed to be taught widely, as I have suggested, I believe they must be started early in life, preferably in the lower elementary grades. Children have several advantages over adults in the process of learning. As I have already mentioned, they do not have a fear of looking foolish. A child, at least before adolescence, trying to learn a complex skill like computer programming often couldn't care less about his or her reputation and simply wants to learn more about these fascinating machines. An adult not already experienced at computer work might spend hours searching in his own mind for an answer while a bright 10-year-old working on the same problem will ponder for only a minute or two before asking someone more knowledgeable for help.

The danger of kids losing their social skills by immersion in computerese may be averted, not aggravated, by an early introduction. This phenomenon is most common in teenagers who are new to computers and still excited about them as an end in themselves. Children who grow up with computers as a fact of life are less likely to jump into them with the same all-consuming fervor I have occasionally witnessed and fallen victim to, but will be more likely to simply accept computers for what they are.

It is a well-known fact that children are particularly adept at learning languages, and I suspect that this ability applies equally to logical thought patterns of the sort one develops after a few hours of playing chess, twisting a Rubik's cube, or programming a computer. In any case, such computer languages as BASIC, PASCAL, and LisP certainly qualify as languages in the way they are learned, though they may have smaller vocabularies and more logical (and hence more easily learnable) grammars than do human languages. Children, therefore, should have less trouble learning both the techniques and the disciplines of computer use and programming than adults would. In addition, they can more fully internalize the logical thought patterns necessary for good programming, so they benefit more from the training.

Conclusion

The invention of computers has caused irrevocable changes in human society, and within 40 years will bring about changes we cannot conceive of today. Even if it were desirable to eliminate them from our lives, we cannot reject them now without crippling civilization for decades. Obviously, therefore, we must adapt to them: learn why they work, learn how to use them, and take advantage of their growth by learning from them. Computers are not "newfangled" things that will go out of style in 10 years. If we do not teach our kids to adapt to the rapid changes the next 40 years will bring, we are making them maladjusted to their world from the start.

It Started with Games

CHRISTOPHER KARNES

My interest in computers began about 4 years ago after meeting a graduate student who was working on his dissertation. He had acquired the use of a computer terminal and invited me over to play some computer games. I was fascinated by what the computer could do. After that my interest in computers was sustained by an occasional book or magazine from the university library. In addition, my interest in electronics was fostered by working with several Radio Shack kits.

Then, suddenly, from out of thin air, my parents gave me a model 1 TRS-80 microcomputer with level 1 BASIC and 4K memory as a Christmas gift. This was about 2 years ago and came as quite a surprise to me.

For the next week, I practically locked myself in my room and worked through the entire level 1 BASIC manual. I had some previous experience with TRS-80's so the first few chapters were relatively easy. For the next couple of months, my time was spent writing BASIC game programs, and reading more about BASIC and computers in general.

The computer even helped me in my school work. I could write programs to quiz myself on spelling words and science and history terms. I even integrated the computer into an independent study project that was required in my school's gifted program. The topic of the project was focused on teaching elementary school children TRS-80 level 1 BASIC. The project was completed with much success.

That summer I began to realize the limitations of the computer hardware I possessed and how the hardware was limiting my software production. So, toward the end of the summer I upgraded my computer system's memory to 16K and language capacity to level 2 BASIC. This brought my computer system closer to the standards I desired.

During the same summer, I attended a computer camp on the University of Southern Mississippi campus. This was quite an experience both educationally and socially for it introduced me to some friends throughout the state that I still call on when in need of programming ideas.

The next year, when I entered the tenth grade, I already knew what my first independent study project was going to be. I wanted to do something with a language other than BASIC. I chose FORTRAN, a language that I had been introduced to at the summer computer camp. But just learning FORTRAN wasn't enough. I had to have something to show for 9 weeks of work. So I decided to write a game in FORTRAN. The game was to be entitled FANTASY and was to be based on the popular role-playing game Dungeons and Dragons.

Then the question arose as to what type of resources should be used. The university library had several books on FORTRAN, but that wasn't enough for me. The best resource I could think of was a university course. I could learn about FORTRAN and get college credit at the same time. I decided to do it, and it wasn't difficult at all. Some of the concepts I used in my game program would later be taught in the FORTRAN course and some of the concepts taught in the FORTRAN course I would later use in my program.

At the end of the 9 weeks, I had a fantastic 1000 + line game program to submit as well as numerous programs completed in my FORTRAN course, and I still had half of the fall semester to complete my FORTRAN course, which I did successfully. It was one of the most challenging experiences in my life. In addition, it helped me in many ways besides just learning FORTRAN. My grades in school rose to new heights, I developed better study habits, and my interest in computers was strengthened.

I enjoyed my first college class so much that I enrolled in an advanced FORTRAN course during the spring semester. With the new course came two new independent study projects. The first was a study on whether the season of birth affects the level of a person's intelligence.

For this project, I used a statistics package available on the University computer system. The data were collected from several school districts on children currently enrolled in gifted programs. This project not only gave me a strong knowledge of statistics and research techniques, but gave me a first place trophy in the State Science Fair in the category of Behavioral and Social Sciences.

I wanted my final independent study project to be spectacular and challenging so I decided to develop my own application language to be implemented on the TRS-80. The language was to be used to utilize fully the TRS-80's limited graphics capabilities and was to use many of the best features of BASIC, PASCAL, FORTRAN and COBOL. The language was to be called GRAPHIX and was to be divided into relocatable modules that could be called upon whenever needed.

The project ran into some problems during the middle of the 9 weeks mainly due to a major hardware malfunction and is currently in the process of completion. I am hoping to use this project as my 1982 Science Fair entry.

Currently, as of the summer of 1981, I am assisting with a 2 week summer program for the gifted at Mississippi State University in which 1 week is devoted to working with computers. I am also trying to get FANTASY, along with several other games written for the TRS-80, published as well as completing GRAPHIX. I am now enrolled in two new courses at the University, one on COBOL and the other on BASIC. My primary interest now is computer graphics and I am saving money to purchase a high quality computer graphics board to interface my TRS-80 which I recently upgraded by adding a variety of peripherals.

I've kept up an extensive computer library which I have supplemented with subscriptions to *Creative Computing* and *80-microcomputer*, as well as many books and textbooks. I also make extensive use of the university library which carries many journals and magazines as well as a large assortment of books.

For undergraduate school, I plan to attend the University of Southern Mississippi or another university depending on scholarship availability. For graduate study, the Harvard Institute of Computer Graphics is a possibility.

I look positively at a career in computer graphics which combines my interests in computers with my interests in the arts.

Computers can be used with great success by all of today's youth because computer programming can be fascinating as well as a great creative and intellectual challenge.

Christopher J. Karnes is a student at Hattiesburg High School and the University of Southern Mississippi.

Two Programs from a Young Eighth-Grader

SAFI BAHCALL

Introduction by Harold Nelson

What would happen if young children were given free access to a personal computer to use at their pleasure—not just to play games and run prepared programs, but to use when and where they wanted for whatever purpose they saw fit?

The accompanying programs are one example of what can happen and their author represents the kind of young self-taught computer scientist that can result.

Eleven-year-old Safi (a nickname for Assaf) Bahcall is in the eighth grade, having skipped fifth grade, at his local middle school in Princeton, New Jersey. He is now completing tenth-grade geometry, already having completed Algebra I. He says, "I do well in math, even though science is my favorite subject." He likes physics, though all areas of science interest him.

"I like computers and have spent a lot of time with them," he admits—a lot of time in the past two years, that is. He began working on computers by simply going into his local Radio Shack store "every afternoon and just messing around with the TRS-80 there." This went on for some time while he taught himself BASIC from several TRS-80 books.

Safi's father (who is employed at the Institute for Advanced Study at Princeton) then got a Radio Shack TRS-80 Level I computer with 4 K bytes of memory. Safi, with advice from a friend of his father, quickly added an expansion interface, increased the memory to 16 K bytes, and added Level II BASIC.

He also added an RS-232C serial interface and modem (modulator/demodulator) which allowed him to use the telephone to tie into the Princeton University computer.

When he hooked up to the Princeton computer, he discovered that their terminal program allowed him to input data but not to display it on his video terminal. So, with a friend, he simply edited their terminal program. Now he can see what he is typing in. Whenever he does tie into the Princeton system, he usually has something specific to do. For example, he prepared the following programs and descriptions using the text-editing capabilities of the Princeton computer.

There are two Apple computers at Safi's school, but he says they are mainly used for games. He has, however, helped some teachers with programs for algebra and science. He has also written a program for the TRS-80 called Dog Race that helps his younger brother practice addition. The program:

"asks you your name and how many people you want to play. You can play against the computer, and you can play against whatever level of difficulty you want the computer to be. Then what happens is, on the screen, you see two dogs that begin at the left edge and are labled *Brian* (or whatever your name happens to be) and *Computer*, or *Brian* and *Joe* if two people are playing. Then the computer asks the first person a problem and, if he gets it right, his dog starts moving so many spaces toward the right edge of the screen. Next, the second person is asked a different question and if he gets it right his dog moves over; or if it's the computer, depending on the level of difficulty, there is a certain chance that it will make or miss the problem."

Safi has written another program to help his nine-year-old brother with his weekly spelling words. But his biggest *job* to date has been working for a person in the Astronomy section of the Institute for Advanced Study. He wrote a fairly long program that used the Monte Carlo method (a statistical technique) to test the virial theorem (a theory from the branch of physics known as statistical mechanics). The person Safi was working with gave him the equations, but Safi devised the way to program them. "I did it for him," he says proudly, "and I got $120 from the Institute for doing that one job. I am currently working on another version of the program."

Safi is saving his money to get a printer. At present, he still has to go to the Radio Shack store with tapes of his programs to get listings.

If you are wondering why a store would let a child come in and start using computer equipment, the reason might be that he helps sell computers. For example, when he went to get a listing of his Dog Race program, some people came over and their children started trying out the game. In fact, the children sat there and played the game all the way through. As a result of this, the parents were interested in getting a personal computer of their own.

When asked about his plans for the future. Safi says, "I've been thinking about something with computers. I've already made $250." (He has now published two more programs.)

From *onComputing*, Volume 2, Number 3, pages 38–42. Copyright © 1980 by onComputing, Inc. All rights reserved. Reprinted with permission.

Any Old Base

Here is a once-in-a-lifetime opportunity to acquire a simple program that converts numbers from one mathematical base to another mathematical base. My program converts a number expressed in any number base from two thru ten to a decimal number (base ten). The first version of my program in listing 1 converts an eight-character binary (base two) number into decimal form.

Before going any further, it is necessary to understand how to calculate decimal numbers from numbers in different bases and how to construct bases (not the military kind, but the kind you use in math). If you are talking to someone and you say, "I bought 120 eggs," you're referring to 120, base ten. Each digit in the 120 (1,2,0) stands for a certain amount of a certain amount. This means that this number stands for 0 ones (0×10^0), 2 tens (2×10^1), and 1 hundred (1×10^2). In base ten, you read the number from right to left, starting with the amounts of $10^0(0)$, $10^1(2)$, $10^2(1)$, etc. It's the same in other number bases, except that instead of the ten you use the other base number. In the same way, you have in base five some number of 5^0, 5^1, 5^2, etc. Let's say you have a 131 in base five. That means 1 one (1×5^0), 3 fives (3×5^1), and 1 twenty-five (1×5^2). Using your pocket calculator, you find that this is 41 in decimal form.

This is the explanation of the program in listing 1. There is just one in-

Listing 1

```
10  CLS
20  REM    ************************************************
30  REM    BINARY TO DECIMAL, BY SAFI BAHCALL
40  REM    CONVERTS AN 8 CHARACTER BINARY TO
50  REM    DECIMAL.
60  REM    ************************************************
100  INPUT"BINARY NUMBER ";A$
110  IF LEN (A$   )< >8 THEN 100
120  FOR I=8 TO 1 STEP -1 : T=T+1
130  A(T)=VAL(MID$(A$,I,1))
140  IF A(T) <0 OR A(T)>1 THEN 100
150  NEXT I
160  REM        * * CALCULATE DECIMAL * *
170  FOR I=0 TO 8
180    DEC=DEC+(   A(I)*(2↑(I-1))   )
190  NEXT I
200  ?"DECIMAL = ";DEC
210  ?:?:INPUT"ANOTHER  CONVERSION  ";W$  :IF  LEFT
      (W$,1)="N"THEN END
220  END
```

Run of Listing 1

```
BINARY NUMBER ?11111001
DECIMAL = 249

ANOTHER CONVERSION ?Y

BINARY NUMBER ?01011000
DECIMAL = 88

ANOTHER CONVERSION ?N
READY
>
```

put to the program and that is an eight-character binary numeral (base two). The eight-character binary numeral is a string called A$. The program stores each character of the A$ string in an array called A. Lines 110 thru 150 check to see that the number is in binary form and store the character in array A. Lines 170 thru 190 calculate the decimal value. The way the program does this is by multiplying the first character on the right by 2^0, the second by 2^1, etc, then adding the results. The characters in the array are stored from right to left. Line 200 prints out the decimal value. Line 210 asks you for another binary number.

There are many things you can vary. You can change the length of the binary numeral (which I made eight in listing 1) by deleting line 110 and changing the eight in line 170 to the length you desire. If you wanted a number ten characters or more, you would have to DIMension array A to your number.

All these changes plus another are included in listing 2. In listing 2, the input is the base and the number. The program now checks whether or not the number is within limits of the base. The restrictions are that the number be less than 100 characters long and the base be two thru ten. These are the main differences between listing 1 and listing 2. V is assigned the length of your number and B the base; all the other variables are the same as in listing 1. The calculational technique remains the same, but the 2 in line 180 of listing 1 is replaced by the base (B).

To sum it up, if you are a base fanatic or want to impress your friends, use these programs.

Listing 2

```
10  REM **********************************************
20  REM    PROGRAM BY SAFI BAHCALL TO TRANSFORM ANY
30  REM    NUMBER IN BASES 2-10 TO DECIMAL.
40  REM **********************************************
50  DIM A(100)
60  CLS
70  INPUT"BASE ";B
80  IF B<2 OR B>10 THEN 70
100  INPUT"NUMBER ";A$
110  V=LEN(A$)
120  FOR I=V TO 1 STEP -1: T=T+1
130  A(T)=VAL(MID$(A$,I,1))
140  IF A(T)<0 OR A(T)>=B THEN 100
150  NEXT I
160  REM      * * CALCULATIONS * *
170  FOR I=0 TO V
180  DEC=DEC+( A(I)*(B↑ (I-1)) )
190  NEXT I
200  ?"DECIMAL EQUIVALENT = ";DEC
210  ?:?:INPUT"ANOTHER   SIMULATION   ";W$:IF
     LEFT$(W$,1)="N" THEN END
220  END
```

Run of Listing 2

```
BASE ? 3
NUMBER ? 22
DECIMAL EQUIVALENT = 8

ANOTHER SIMULATION ? N
READY
>
```

How Much Have You Lost or Gained?

Gold is going up, silver is going down, and all the commodities are jumping around. If you have something invested in commodities, this program will interest you. It works on a TRS-80 Level II computer, and after you input how much money you have invested, how many ounces (or whatever standard unit) you bought, and tne going rate per unit for your commodity, it will tell you the net amount you have gained or lost. In the net I include, for example, a three percent investment fee and a one percent selling fee, so if you bought the commodity at the same rate at which you sold it, you lost four percent.

The variables are A *for amount of money invested,* O *for the amount of standard unit, and* RATE *for the current rate.*

Lines 4 thru 100 are for decorative purposes only and are not necessary.

If you cannot remember how much you spent on the commodity, only the going rate at that time and how many units you bought, change line 110 to read:

110 INPUT "RATE (PER STANDARD UNIT) AT WHICH YOU BOUGHT THE COMMODITIES";V

Also, add line 135 to read:

*135 A=V*O*

After this is completed, the computer will ask you the rate at which you bought the commodity. It will derive how much money you spent by multiplying that rate times the number of standard units you obtained .

Line 140 adds the three percent investment fee to the total amount you have spent. Line 150 calculates how much your commodities are worth, and deducts the one percent selling fee. (To change the percentages, change the numbers in these two lines.) Line 160 assigns the difference between the current value of your commodities and the purchase price of your commodities to DI. Line 170 branches off to another part of the program, depending on whether or not you have gained or lost money. Line 300 tells you how much money you have gained; line 180 tells you how much money you have lost. You are limited to a gain or loss of $99,999.99. In case you do not own a Level II TRS-80, the question marks in the listing are shorthand for "PRINT" statements.

If you are a person who just gives your money to a company that can invest it in any way the company decides, or if you are just curious about how the money you have invested in commodities is doing, you can find out using this program.

Listing of Gains/Losses

```
4   '     ..............................................................
5   '       PROGRAM BY SAFI BAHCALL TO
6   '       CALCULATE GAINS/LOSSES IN
7   '       COMMODITIES.
8   '     ..............................................................
10  CLEAR 500: E$="#####.##"
20  ? STRING$(63,131) : ? TAB(17)"* COMMODITY CALCULATOR *"
30  ? STRING$(63,131)
100  '
110  INPUT"AMOUNT OF COMMODITY INVESTED (IN DOLLARS) ";A
120  INPUT"AMOUNT (IN STANDARD UNIT) OF COMMODITY BOUGHT ";O
130  INPUT"CURRENT RATE OF COMMODITY (PER STANDARD UNIT) ";RATE
140  LET A=A*1.03              ' ADD 3% INVESTMENT FEE
150  LET G=(RATE*O)*.99        ' CALCULATE VALUE+1% SELLING FEE
160  LET DI=G-A                ' CALCULATE DIFFERENCE
170  IF DI=0 THEN 300
180  ?:?"YOU LOST";:?USINGE$;-DI;:?" DOLLARS."
190  ?"TAKING INTO ACCOUNT THE 3% INVESTMENT FEE,"
195  ?"AND THE 1% SELLING FEE."
200  ?:INPUT"ANOTHER SIMULATION ";W$:IFLEFT$(W$,1)="N" THEN END
300  ?:?"YOU GAINED ";:?USINGE$;DI;:?" DOLLARS.":GOTO190
999  END
```

Run of Gains/Losses

```
                        * COMMODITY CALCULATOR *

AMOUNT OF COMMODITIES INVESTED (IN DOLLARS) ?16000
AMOUNT (IN STANDARD UNIT) OF COMMODITY BOUGHT ?40
CURRENT RATE OF COMMODITY (PER STANDARD UNIT) ?520

YOU GAINED 4112.00 DOLLARS.
TAKING INTO ACCOUNT THE 3% INVESTMENT FEE,
AND THE 1% SELLING FEE.

ANOTHER SIMULATION?
```

3 Computers at Home

This brief section highlights the value of providing computer experiences at home. In the first article, *Teaching Parents About Using Microcomputers,* Dwyer and Critchfield maintain that the fundamental prerequisite for involving children in creative programming is to have parents explore and be creative themselves. The article outlines sample topics for informal workshops in the art of personal computing. The authors believe that schools should consider offering computer workshops for parents so that they, in turn, can help strengthen the computer experiences of their children.

In his article, *Children and Home Computers: Some Observations on the First Generation,* Bernard Banet, developmental psychologist, curriculum developer, and father of an 8-year-old, reflects on the present and future of home computers and their use with young children. This article was included as food for thought for parents who are making decisions on how to involve their children in activities using home computers.

The final article, *An Apple a Day Keeps a Kid Occupied* by Ron Buszta is a very short piece written by the father of a 4½-year-old. It was included in this collection to show that even very young children find computers stimulating, educational, and fun!

Teaching Parents About Using Microcomputers

THOMAS A. DWYER
MARGOT CRITCHFIELD
University of Pittsburgh

Introduction

Most educators will readily admit the importance of home learning experiences for a child's intellectual development. While schools can do a great deal, the youngster whose parents use language and mathematics with zest in their daily lives has opportunities for incidental learning that nourish the intellect. Just how this mental diet is absorbed is a mystery, and trying to solve this mystery is bound to give learning researchers job security for some years to come. Meanwhile, much can be done.

Most parents try to feed their children the seven basic food groups and fend off junk food. There is also some wisdom accumulating regarding informal learning which can help them do something similar for their children's minds (besides just turning off the TV set).

Buying an encyclopedia or a set of The World's Great Literature for Young People is a good start, but more is needed. Involving the young person in doing something creative—something that requires the use of the abstract symbols of language and mathematics—is vitally important. But it is not always easy to do this.

Enter, the Computer

The significance of the home microcomputer for informal learning is that it provides a whole new range of interesting things for young people to create with the help of their parents—*computer programs.* Making the computer *do* something can be a fascinating intellectual activity, not just for the adult hobbyist who "pushes bits around," but for the young child who creates a simple picture on the computer. And the skills of reading, typing, arithmetic, logic, algebra, vocabulary, etc., which are brought into play during this activity provide a kind of practice quite different from the workbook or drill lesson.

The question is, can parents, without a technical background, learn enough about programming (and about what the computer is and can do) to make this new intellectual tool effective for their children? To say this another way, how do you master a machine that can be used so many ways? Some are simple to deal with (for example, learning to play a preprogrammed game on a microcomputer is no problem). But getting this same machine to do things that go beyond the applications shown in books takes both creativity and know-how.

Short Courses for Adults

To explore the possibility that the "average" adult could meet this kind of challenge and become an inventive computer user, we've recently been developing and teaching a number of informal workshops in personal computing. These have revealed a range of talent and flexibility that is heartening. The syllabus we have developed as a result of working with these adult beginners is quite broad, but we believe that it is particularly relevant to the parent who wishes to act as an informal teacher using a home computer. The fundamental prerequisite for involving your child in creative programming is to do some exploring yourself—to be creative. This means knowing a lot more than how to run a packaged CAI program.

Our workshops have, therefore, been designed as an introduction to personal computing for persons without previous experience in the field who nevertheless want to attack it *creatively.* To put it another way, the workshops are for anyone who wants to get started on the fun and satisfaction of "solo computing."

Learning to go solo with a computer means learning to be in charge—to know not only what the computer can do, but how to make it happen. It's the difference between admiring the wonders of jet flight from a passenger's seat, and moving up front to do a few lazy eights around the sky yourself.

Fully mastering personal computing at that level takes a while, of course, and a five or six week workshop should only be labeled as a start. But it's an important start, and taking a solo approach—even at the beginning—is less difficult than might be suspected.

Course Content

Three core questions seem to be uppermost in the minds of adult students: (a) What *are* microcomputers? (b) How do I choose one wisely? and (c) How do I go about using it for the applications *I* have in mind—including learning in the home?

We've translated these questions into three main goals for the workshop. The first to help students develop some technical familiarity with the microcomputer field, especially as it applies to personal and business computing. The second is to share what we and others have learned about evaluating and buying a personal computer system. The third is to explain (through concrete examples) how to use a microcomputer to its full potential.

There are many specific topics suggested by these general goals, and a few new ones seem to surface each time we teach the course. We've tentatively grouped the topics of interest under seven headings. The headings and some of the topics they include are as follows:

Group 1. The New Look in Computers

What *is* a computer?
Using a personal microcomputer.
Inside microcomputers; the LSI breakthrough.
But is personal computing a good idea?
What to do until the computer arrives; using time sharing.

From *NCC '79 Personal Computing Proceedings,* pages 59–63. Copyright © 1979 by the American Federation of Information Processing Societies. Reprinted with permission.

Group 2. The What and How of Microcomputer Systems

Microcomputer system terminology; sample systems.
Communicating with a microcomputer; peripherals and I/O.
Computer central; the CPU and memory; more on I/O jargon.
Mass memory; the world of megabytes.
The mix and match problem; customized versus packaged systems; examples.

Group 3. Mind Over Machine: Computer Software

Kinds of software; system versus application programs.
An introduction to BASIC.
Extended BASIC.
Structured programming; designing and writing a longer program.

Group 4. Selecting and Buying a Microcomputer

Developing a checklist of your needs; examples.
Applying the checklist; examples.
Shopping for a computer; what's available.
Dealing with change; upward expansion of your system.

Group 5. Using Personal Computers

Some short programs to try.
Learning with the computer.
Computer graphics and games.
Data bases in computing; the computer dating example.
Home finance programs.

Group 6. Microcomputer Business Systems

Can a computer really help a small business? The pros and cons; examples of business programs.
The hardware requirements of business systems.
Software requirements; computer files; disk extended BASIC.
Using off-the-shelf application software.
The argument for customized software; hiring a programmer; documentation and maintenance.
Guaranteeing success; the virtues of patience, redundancy, and pessimism.

Group 7. Planning for the Future

What changes are possible? Probable?
Hardware updates; when to start over.
The future of software; phasing in change.
Keeping informed; sources and strategies.

These topics have been grouped in a "logical" order which is not necessarily the best one for teaching a class. For example, most students prefer learning how to program in BASIC right away and finding out how computer hardware actually works later on. There has also been a growing interest in learning to use all the features of extended BASIC. Most people (including some computer professionals) have no idea of how powerful the next extended BASIC interpreters are. They are particularly flabergasted at the many elegant things possible with the extended BASIC on the classroom demonstration computer we use (a TRS-80 with Microsoft Level II BASIC). Students have been unanimous in agreement that this extended BASIC is not only more powerful in its features, but far easier to "manipulate" than the BASIC on a large time-shared computer they also use. The lesson about the importance of good software gleaned from this experience is notable, especially in a field where the glamour of hardware can be so enticing.

The application interests of students have also been sampled at the beginning of each workshop in order to guide our curriculum development. These interests vary, of course, but two areas that seem to top the list are *structured* games with graphics and *structured* business applications. To respond to these interests, we're developing some new materials in both areas. One (the BABYQ structured "quest" game) is turning out to be an excellent way to transition from elementary to advanced programming in a very short time. It's fun, but it also includes experience with such mathematical ideas as probability, coordinate geometry, Euclidean distance, and the use of data structures.

Other interesting structured programs (and the areas they promote learning in) which we've used are SALESLIP (business math), AIRPLANE (use of matrix transformations to rotate or translate pictures on the screen), GAUSS (solving big linear systems just like the pros), ARROW (trajectory motion based on Newton's Laws), MATHPLOT (painting pretty pictures on the screen that are derived from classical math functions), and a variety of smaller word, puzzle, and "story" games to exercise the use of vocabulary.

There's little doubt in our minds that putting a computer to work in the home as a "solo learning" tool is one of the most exciting educational ideas to come along in years. Helping parents learn how to exploit this idea is something every educational institution ought to consider.

Children and Home Computers: Some Observations on the First Generation

BERNARD BANET
Director of Planning and Development
High/Scope Educational Research Foundation
Ypsilanti, Michigan

As a developmental psychologist, father of an 8-year-old, and curriculum developer, I would like to offer the following thoughts on the present and future of interactive electronic learning systems in the home for preschoolers and children in the elementary grades. In the past two years I have had the opportunity to participate in the development or informal evaluation of a variety of home educational applications of devices ranging from calculator-like game/drill units to personal microcomputer and educationally oriented time sharing systems. In outline form, here are some conclusions I tentatively propose:

1. The capacity of interactive electronic systems to be useful in home learning for young children is currently best seen in instructional games and simulations.
2. Drill-and-practice applications, stripped of gamelike elements, do not survive for long in the home environment without parental insistence. It doesn't take more than a bit of ingenuity, however, to add gamelike elements to even the dullest drill.
3. The widely advertised potential of microcomputers to stimulate interest in computer literacy, programming, and hence in the wider world of math, science, and technology is also real, even for young children; however, it is even more dependent on human support (from parents, siblings, friends, teachers) than the games and practice applications.
4. Creative activities in art and music on home computers are still aesthetically crude but suggest exciting new modes of personal expression.
5. Computer-based activities, whether games or programming, can be an occasion for very positive parent-child and child-child interaction. Computer activities are not unique in this respect, of course, but can be seen as parallel to reading together, playing board games, going on family outings, working on home maintenance, cooking, gardening cooperatively, etc.
6. For young children, computer-based activities of the first generation may be as important as a source of motivation to learn new concepts and skills as they are as direct sources of learning.
7. The recreational potential of microprocessor-based devices can *compete* with, as well as support, use of these devices for learning and problem-solving. A situation parallel to the "personal" use of books and television exists in this respect.
8. The usefulness of computer-based home learning systems will increase *if and when* the following occur:
   a. Effective applications software is developed and distributed:
      - Software is sold in retail stores so that the consumer can sample on the spot and buy off the shelf.
      - Learning software (or "courseware") is designed in interconnected sets as well as in discrete games and activities.
      - Programs are designed to adapt (under user control and/or program control) to the user's degree of skill mastery and familiarity with a particular program.
      - Learning software interfaces with audiovisual media, print, and human resources rather than trying to emulate these when it cannot effectively do so.

   b. Hardware/Software configurations are friendlier to the user as:
      - Cassettes are replaced by mass storage media that permit rapid random access to a program library or data base and permit file-updating and record keeping in a reasonable manner.
      - Software becomes more compatible or transportable across systems.
      - Random-access audio output for voice messages is feasible and affordable. This is important for children who do not yet read with ease. It is hoped that the quality of human speech can be preserved in such systems better than with today's voice synthesizers. Digital encoding of real speech (as on a videodisc or Speak & Spell chip) may be the answer.
      - Such "extras" as lower case character generators, user-oriented graphics languages, text editors, printers, and communications interfaces become commonplace on home systems.

   c. Software is documented, catalogued, and critically reviewed.

There is obviously yet much to be done to make personal computing into the powerful educational tool it can be for learners of all ages.

From *NCC '79 Personal Computing Proceedings,* pages 62–63. Copyright © 1979 by the American Federation of Information Processing Societies. Reprinted with permission.

An Apple a Day Keeps a Kid Occupied

The sines of the father are visited on the son.

Ron Buszta
Oakwood Dr.
New Fairfield CT 06810

Never did I think, when I bought my Apple II, that I would have to fight with my 4½ year-old son for its use. But that's exactly what has happened. My son, Jeffrey, who is still in nursery school, has been inseparable from that machine since I bought it last summer.

He doesn't read yet, but he does recognize some words, such as Jeff, Mom and Dad. I wrote his name on a diskette containing some graphics games so he could identify it when he wanted to play some of those Apple programs. After the first couple of times he played Dragon Maze and Breakout, he wanted to push the diskette into the drive by himself.

He next wanted to start up the Apple from scratch without any assistance from me or from my wife (he's a rather independent youngster!). I explained that it wasn't an easy procedure—you had to turn on the monitor, press the CTRL key down and at the same time press the letter B, repeat the pressing of CTRL and B until the prompt (>) character appeared and, finally, type PR#7 to tell the operating system that the disk was in slot number 7.

I was thoroughly convinced that I had confused the inner workings of his little brain and that he would lose interest and go back to watching Woody Woodpecker cartoons. As I sat down to have myself a good game of Breakout, I heard him say, "Could you show me that again, Daddy?"

He thought he could remember that whole sequence of events and turn on the Apple himself! Realizing he is a very determined little boy, I didn't argue; but I proceeded to tell him again how to communicate with the Apple. After two or three more explanations, Jeffrey had it all but mastered—turn on the Apple, turn on the monitor, press CTRL-B twice and type PR#7. A broad smile crossed his face when he heard the pleasing whirr of the disk drive.

"Now what, Daddy?" he asked. I told him that he couldn't do much more since he couldn't read, but that I would load and run whatever programs he wanted to play. Well, after I did that for a while, naturally he wanted to run the programs. "R-U-N" I would say to him to get a program to run.

It didn't take long until he remembered how to spell RUN or that CTRL-C would stop a program and that the backward arrow would erase a letter that was pressed by mistake.

After a couple of weeks, Jeffrey was typing CATALOG for a list of all the programs on disk (more for the joy of hearing the disk drive run and seeing the pretty red light flash on than for all the info going to the screen) and LIST so he could make all those letters go up the screen. He learned that while playing Dragon Maze, typing R, L, U and D would move the little squiggly character to the right, to the left, up or down in an effort to elude the dragon and gain access to the doorway he guarded. Just remembering what those letters represented, I think, was a great accomplishment and well worth the price of the Apple.

But the learning process for Jeffrey and me did not stop there. While I was learning BASIC from listing programs and from reading the Apple documentation, Jeffrey was learning to recognize RIGHT, LEFT, DOWN, LIST, RUN and CATALOG.

Jeffrey has just turned five years old, and I expect any time now that he'll be teaching me about the SIN and COSINE functions or how to write a program to produce the Fibonacci sequence. ■

From *Microcomputing,* March 1980, page 172. Copyright © 1980 by Kilobaud Microcomputing. All rights reserved. Reprinted with permission.

4 Programs in Action

Material for this section was chosen to reflect the variety of programs operating in schools today. Children and youth of all ages are making computer connections in programs that range from total computer availability in the regular classroom to Saturday enrichment programs. Each of the programs described is different and each can serve as a model. There are possibilities here for every level of commitment from districtwide, to single classroom, and from mainstream settings to specialized programs.

In *Microcomputers for Gifted Microtots,* Ann Doorly describes a mathematics enrichment program for gifted children in grades one through four. Because only one computer is available, it is set up as part of an "interest center." Children learn how to key in programs in BASIC and how to solve programming problems. According to Doorly, "these learners not only possess the ability to apply the necessary logical thinking and mathematic skills to operate the computer but also understand the mechanics of what is taking place."

Sally Greenwood Larsen works with gifted and talented third and fourth grade students. She teaches BASIC programming as part of their mathematics class. *Kids and Computers: The Future Is Today* describes the step-by-step procedures Larsen uses to introduce the computer to her students. Larsen, like Doorly, has only one computer in her classroom. She finds that by scheduling children in pairs, everyone can have at least one practice period a week.

Micros "GOTO" School by Donald T. Piele is a report of a pilot project in which an Apple II microcomputer was placed in a sixth grade classroom for 8 weeks for the purpose of developing logical thinking skills. The article includes sample practice exercises in drawing color graphic designs.

The Hampton City Schools Computer Program, under the direction of Nedra Harkavy, has been providing computer experiences to its children at a district wide level for several years. Among the many uses, computers help students create poetry and solve simulated real-life problems.

The Paducah Tilghman High School "Chemics" Program by Steve Johnston provides a candid discussion of how a high school restructured its courses in chemistry, physics, and computer science to provide a comprehensive program for its advanced students. This article provides some practical insights into how a school can combine several courses to provide more challenging programs for gifted students.

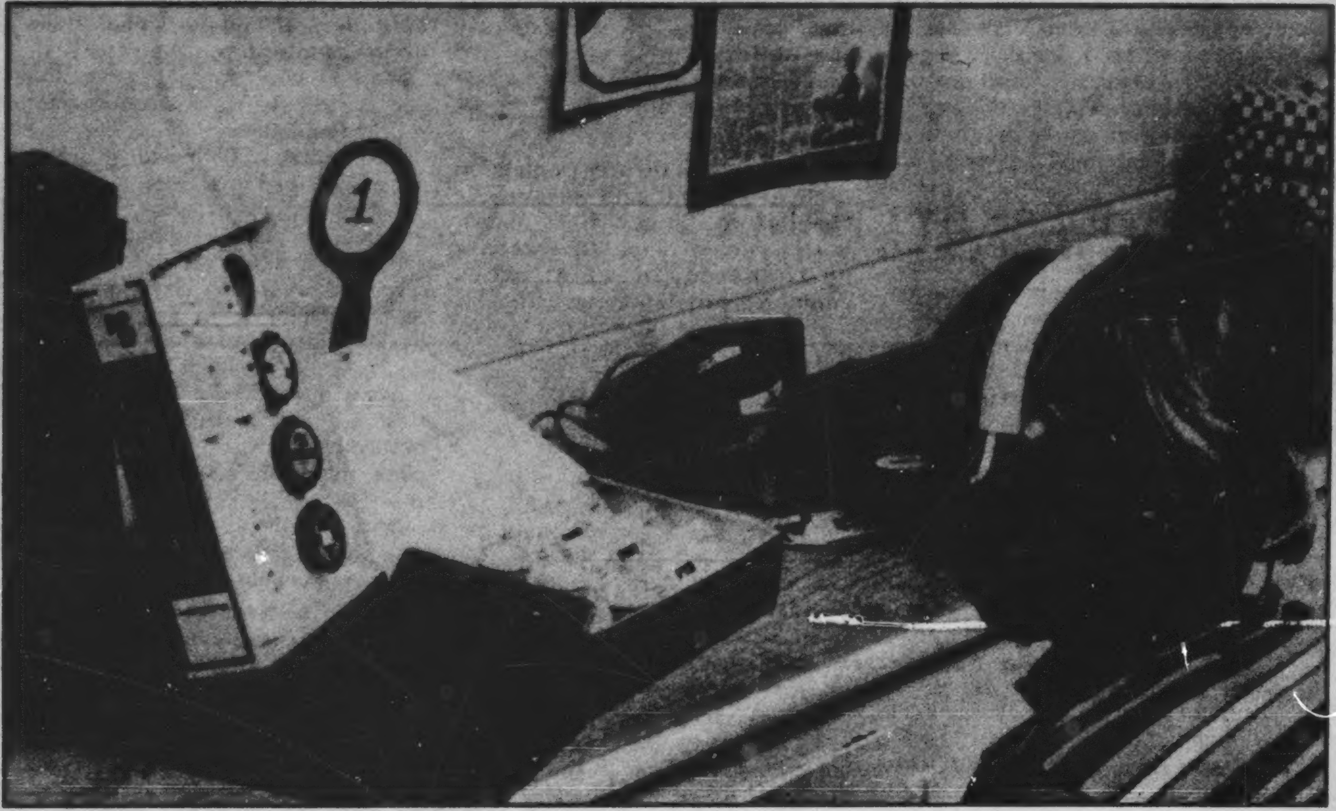
Computing at a New Public High School for Gifted Students by Steve Davis and Phyllis S. Frothingham provides a detailed description of how computers are being used at the North Carolina School of Science and Mathematics, a special public high school for 11th and 12th grade gifted and talented students. The school provides experiences with a variety of computers which are integrated into the total school program.

If it is not possible to develop computer capabilities at the public school level, there are other alternatives. The next two articles describe Saturday enrichment programs. *Computers . . . Are All Dinosaurs Dead?* by Douglas Glover describes a program for youngsters from 8 to 18 years old at the University of South Alabama. He describes the program and then gives some advice about getting computer programs started.

Another Saturday program for gifted and talented children is operated at the Talcott Mountain Science Center in Avon, Connecticut. The article by Daniel Barstow describes how students work on independent projects including satellite tracking, meteorology, geology, and chronobiology. Students often begin by using existing software and then advance to writing their own programs. At the Talcott Mountain Science Center computers have been an integral part of the program for ten years; they are an essential tool for exploration.

The final article in this section, *Computeronics: A Course in Computer Literacy* by Pristen Bird, describes a program developed by the Gifted Child Project in Tallahassee, Florida. This program has been selected as an exemplary program by the Joint Dissemination Review Panel of the National Diffusion Network Division, U.S. Department of Education and is listed in the 1981–1982 National Diffusion Network catalog. Computeronics is a 35 to 40 hour course in computer programming, problem solving, and literacy. The course is divided into two modules: Problem Solving with Computers and Computers in Society. The project also provides inservice training.

Microcomputers For Gifted Microtots



by Ann Doorly

COMPUTER USAGE PROvides excellent horizontal enrichment for primary children who are gifted in mathematics. It presents them with the opportunity to apply the higher level cognitive skills which they are able to grasp so readily. As computers continue to dominate our society, computer literacy, is becoming the fourth "R" in education. If today's students are to be tomorrow's creative producers, it is our responsibility as educators to provide them with the necessary time, space, and resources to assist them in the development of their individual talents and abilities.

Issues which should be addressed in computer science programs for gifted students are: Why is it a differentiated enrichment experience for primary gifted students? Which children qualify for participation in such programs? What types of programs can be developed to teach computer programming to gifted students in grades one through four?

From G/C/T, September/October, 1980, Number 14, pages 62–64. Reproduced with permission of G/C/T, P.O. Box 66654, Mobile, AL 36660.

Why should gifted primary children be involved in computer programming? These children have the ability to think abstractly and to utilize the mental processes associated with higher levels of learning. The computer is a blank slate which requires very specific logical instructions organized into a program in order to function. Gifted children are able to see this process as a mirroring of the human mind and to employ the necessary logical sequencing, critical thinking, and mathematical concepts to design the solutions to real problems. A tremendous amount of task commitment is necessary to persevere and overcome the tedious task of learning to program. Many average students lose interest at this point. The elementary math curriculum becomes more meaningful as students apply the concepts to an exciting relevant task. The versatility of the computer allows each individual student to use the tool to pursue his or her individual interests. Simulation programs challenge the child by allowing him or her to experience the decision making process and the corresponding consequences in life like problem situations. The mechanics of the computer provide the scientifically talented child with the ultimate challenge in electrical circuitry. Since there is usually only one computer terminal and several children working together at one time, cooperative interactive skills are necessary to work together as a team.

One specific school-wide goal should be to prevent "computer phobia", from which too many adults now suffer. For example, a group of secretaries refused to enroll in a free computer programming course simply because they didn't want to use computers. Participation would have enabled them to utilize a computerized text editing system which would have cut their work load in half. When children work with the computer, they not only realize that it presents no threat but often prefer working with it to going out for recess. I experienced "computer phobia" when I accepted my current position as a teacher in the gifted program and was given a Radio Shack TRS-80 microcomputer to use in my primary math enrichment programs. Family and friends who worked with computers assured me that programming was "easy". Having been a first grade classroom teacher for four years, my background included no exposure to computers. My hesitation persisted until a student's parent who is a computer consultant offered to do a series of lessons with the children. As she explained the programming process, I realized it was as simple as everyone had promised. My enthusiasm grew as I discovered materials, activities, and resources with which to expand the unit.

From a total district population of five hundred and seventy-six children in grades K-4, almost twenty percent of the students have participated in eight week computer programming units. Six participated from grade one, twenty-two from grade two, twenty-four from grade three, and fifty-seven from grade four. We have recently purchased with grant money TRS-80s for each of the elementary schools and two additional microcomputers for the middle school.

I plan to conduct inservice workshops for the staffs, to teach whole class introductory lessons, and to encourage the children who have learned programming to help other children run the various available programs. This will eliminate any "computer phobia" on the part of staff and students and encourage maximum usage by all. Many supplemental and individualized programs are available which provide immediate feedback for slow learners as well as drill and enrichment activities for average and gifted students. It provides another vehicle for reinforcement. The computer is now being used by students during all recesses and free periods. This time has been made available by compacting of curriculum and by students' rapid completion of classroom assignments. These "mini-programmers" range from age six to ten and many of them are writing their own programs. These programs will be compiled into a book which will be used by all children to program the computer. The exceptional programs will be submitted to computer magazines for possible publication. Most important, these children will be ready for more sophisticated programming at the middle school, where four Level II 4-K TRS 80s are available and the high school, which has terminals hooked up to the University of Connecticut IBM System 370. Since the children are ready for such instruction, advanced exposure will enable them to begin their creative productive careers at an earlier age.

Students involved in the computer unit are first through fourth graders who have been recommended by a classroom teacher or parent or who have demonstrated high mathematical aptitude and computational skills on the Stanford Achievement Test. Children who do not have the necessary ability and task commitment usually lose interest and ask to "drop out". Children are grouped according to grade level and the "pull-out" enrichment groups allow the children to interact with peers from other classes with similar strengths and interests. Since there is only one computer terminal and up to eight children working with it at one time, much cooperative team work is required on the part of the children. To allow more time for independent programming beyond class time, the children work in pairs. This makes it a social activity and provides practice in working with others. They are also able to diverge at this point from the group and apply their programming skills to their own specific interest areas.

The unit is introduced by showing the filmstrip *Computers: From Pebbles to Programs* (New York: Guidance Associates, 1975) which narrates the history, development, types, parts, and uses of computers. We then count translate, add and subtract (first and second graders), and multiply and divide (third and fourth graders) in the binary number system. It always seems to develop into a contest to see who can count the highest and to date the record is 11111010000 in base two or 2,000 in base ten! The children enjoy the challenge provided in thinking in another system. Basic electrical circuitry is then discussed and the relationship between the on and off states on the panels and the numbers or symbols they represent.

An "interest center" is set up which contains: the computer; sample input/output devices and media; sample first, second, and third generation circuitry; computer task cards; collage posters of computer types, parts, uses, flow charting, and vocabulary; books; magazines; and audiovisual materials. A typewriter and typing manual are available for the children to develop typing skills.

Now the actual programming in BASIC language begins. The children are often surprised to discover that the computer is actually a "dummy" and is only capable of doing exactly what we program it to do. We begin with simple arithmetic problems to demonstrate how algebraic equations (good old new math number sentences!) and variables are used to represent numbers. We develop a flow chart for each program to organize the commands into a logical sequence. Each step in the flow chart corresponds to a statement in the BASIC language. The statements are assigned numerical addresses and keyed into the computer, with the children taking turns typing in the statements. At this point excitement is at its peak, but more task commitment is required to debug the program! The BASIC trouble-shooting facilities "what" and "how" help the children locate errors to correct them. Finally, the program is "run" and each of the children is given the opportunity to run it independently. They then begin to see different alternative capabilities and begin to modify the program by changing statements. The memory can be checked to determine the number of "*bytes*" the computer has used to store and run the program.

During the course of the unit, the children learn continuous looping, conditional branching (in which the computer makes decisions based on the comparisons using the , , and the = signs), data input and output, the use of random numbers and graphics (which is accomplished by lighting points on the screen which is similar to a cartesian plane).

Pretests given at the beginning of fourth grade to students who had completed the unit as third graders showed that retention is poor except in cases where children had continuous access to a computer on which to use their programming skills. Since a comprehensive review is necessary, we apply the arithmetic operations to various math and science problems. For example a basic multiplication program can be modified to a "rate times time equals distance" problem and thus to determine the speed at which a given point at a given latitude on the earth is rotating. This provides the review and the application to real problems maintains a high interest level.

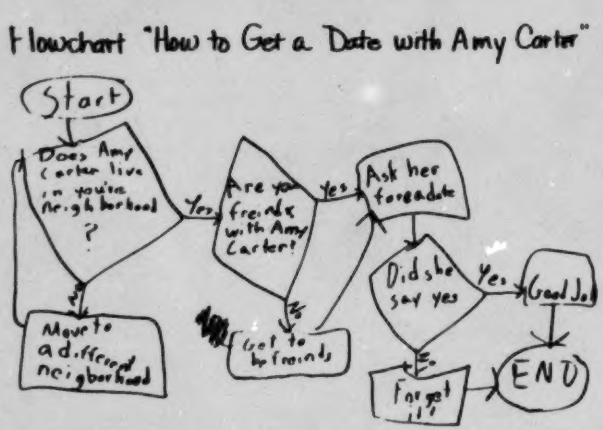
Those children who have the willingness to continue now become involved in independent investigations based on their particular interests. Activities will go in as many different directions as there are turned on excited children. One child may wish to design and build a computer while another may elect to write a program for a new game. At this point, the teacher assumes the role of "manager". The child will need assistance in focusing his or her interest into a solvable problem or specific project. The teacher can't be an expert in all areas so must locate other resources, materials, and methodological techniques to assist the child in pursuing his or her interest through independent investigations or projects. The community, students, parents, teaching staffs, university personnel, and local clubs and organizations are excellent sources for willing human resources. Usually when a person has expertise in a given area, he is enthusiastically willing to share it! Local, state, and college libraries, corporations, government agencies, and universities can provide an abundance of additional materials and information. Finally, upon completion, the child's product should be shared with an appreciative audience.

In conclusion, I wish to again stress the appropriateness of computer science for these young mathematicians and scientists. These learners not only possess the ability to apply the necessary logical thinking and mathematic skills to operate the computer but also understand the mechanics of what is taking place. The applications of the once isolated K-4 math concepts to a relevant endeavor add new challenge and excitement to their elementary school experience. They also gain a head start in preparing for their careers, many of which are sure to involve computers.

Ms. Doorly *recently completed work toward her Master's degree in the Teaching the Talented Program at the University of Connecticut. She coordinates and teaches in the SAGE (Shared Approach to Gifted Education) Program in Mansfield, CT. This article is her first for G/C/T.*

Kids and Computers: The Future Is Today

Sally Greenwood Larsen


Flowchart "How to Get a Date with Amy Carter"
Start
Does Amy Carter live in you're neighborhood ?
Yes
No
Move to a different neigborhood
Are you freinds with Amy Carter!
Yes
No
Get to be freinds
Ask her for a date
Did she say yes
Yes
No
Good Job
Forget it
END

Computer dealers in Racine, Wisconsin aren't surprised any more when an eight year old in overalls, a striped T-shirt, and bumper tennis shoes strolls into the store, sits down at the keyboard of a TRS-80, and writes a computer program for the drive-up window of a McDonald's restaurant. Or a program to print out all the even numbers from 1 to 100, in four columns, with a half second delay in between. Or a graphics program for a birthday cake, complete with blinking candles.

These third and fourth graders are students at the Jefferson Lighthouse School, a program for gifted and talented children, where I teach BASIC programming as part of their mathematics classes. With an average of 45 minutes a week of group instruction over the past year, students in these classes have mastered the concepts covered in introductory college courses in programming, and have become adept in the use of the school's microcomputer. In an area where a strong math background has been a customary prerequisite, it is amazing to realize that most of these children are just learning to multiply and divide!

Our microcomputer is in use from the minute school opens in the morning until the last child leaves in the afternoon, and the only games they play on the machine are those they have written themselves. It's exciting to watch, and even more fun to teach. For a society which will be computerized beyond our imagination by the time these children are adults, it is sad to see the majority of elementary schools using their affordable, portable microcomputers only for computer-assisted drillwork, or playing guessing games. Children can easily learn to write their own programs. However, the elementary teacher who wants to teach programming to kids faces the problem of finding materials which take into account their conceptual development and reading levels. Finding relevant examples is also not an easy task. Physics problems and checkbook balancing simply will not do.

I dealt with this problem by writing my own materials, building the lessons on the following framework:

1. What is a computer?
   A perspective on why computers came to be, what they are used for, and what kinds of jobs they are and are not capable of doing.
2. How does a computer carry out your instructions to get a job done?
   An explanation of simple linear logic, using flowcharts.
3. How do you communicate your instructions to the computer?
   The BASIC language.
4. How do you put together a program which is both efficient and creative?
5. What uses do we make of computers? What new uses can we invent or forecast?

Children have funny notions about machines in general, and especially computers. They need a mental picture of what goes on inside a computer, and its purpose.

What is a Computer?

When a caveman had work to do, he had no tools or machines to help him. He had to do it all by himself.

Man has since invented many tools to help him with his work.

Instead of pounding with his hands, he now uses a hammer. The hammer lets him pound harder and longer than he could with his hands alone.

Man invented the telescope so that he could see farther into space. He can now see stars he did not know existed before he had the telescope to help his eyes.

Using his brain, man can remember information and solve problems.

Man wanted to invent a tool so that he could extend the use of his brain, so he invented the COMPUTER.

Just as a hammer can't do work without a person to hold it, a computer cannot do work without a person to run it, and tell it what to do. This person is called a PROGRAMMER.

Even the best hammer cannot do all the different things our hands can do.

And even the best computer cannot do everying our brains can do.

The author does teaching, curriculum development, and consulting in the field of Gifted Education. Her specialties are math, science, and computer science.

Sally Greenwood Larsen, 1643 LaSalle St., Racine, WI 53404. Photos by Jon Bolton and David Ahl.

From *Creative Computing,* Volume 5, Number 9, pages 58–60. Copyright © 1979 by Creative Computing. All rights reserved. Reprinted with permission.

Kids can't...

A computer cannot feel emotion. It cannot feel happy or sad, as we can.

A computer can't combine ideas like our brain can. It can't put two ideas together and take the best parts of each one to make a brand new idea.

BUT...a computer can do some of the simpler jobs our brains can do. And it can do some of them even faster than our brains can!

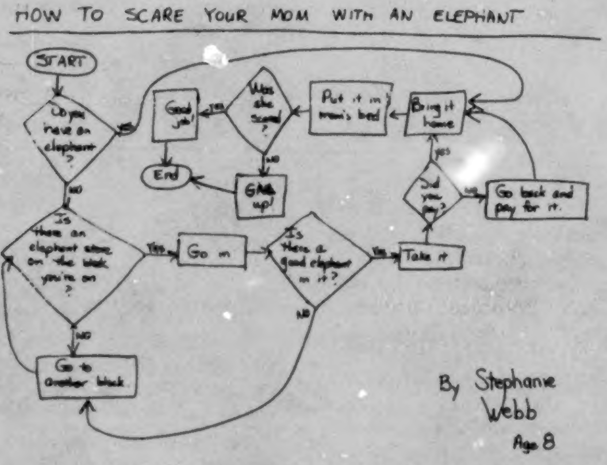
A computer can remember many more things than most of us can with just our brain, especially things like long lists of names or numbers. Information stored in a computer is called DATA.

A computer can compare data, to see if one thing is bigger than another, or smaller, or the same. It can also put things in order.

A computer can sort lots of pieces of information and put together the things that are alike.

And a computer can recall the information a computer programmer wants, and print it out for him on a video screen or a sheet of paper.*

Once the children have an overview of the function of a computer, they need to see graphically how a computer program breaks down a task into small steps, and progresses in linear fashion from one step to another. Simple flowcharts, showing an activity with which the children are familiar, produce an easy to digest and sometimes hilarious picture:



Now the children are ready to see that the way in which we communicate with a computer must be standardized, and we need a particular language for this purpose, and a set of rules for typing in statements on the machine. After group instruction in operating the keyboard, and statements dealing with the execution of the program, such as CLS, BREAK, NEW, LIST, RUN and END, the children can do a surprising amount of experimentation with the simple PRINT statement and its variations.

```
PRINT "My name is Jerry DeMaio. I
   love computers."
PRINT "RIGHT"
PRINT "LEFT"
PRINT
PRINT "8-4"
PRINT 8-4
PRINT "*?&$#%)"
```

It is essential to schedule all the children on the computer at least once a week, so they practice what is learned in group instruction. Pairs of children seem to work best, for one child alone gets "stuck" too often, and three or more argue over who will type on the console.

Allowing the children, especially the youngest groups, to use prepared game programs at this point is a serious mistake. It kills their desire to put in the effort required to learn BASIC since canned programs are so much less work. It is interesting to note that when game tapes are made available to children who are already fluent programmers, they will typically play them once, LIST them and see if they can pick up any programming tricks and then abandon them.

Worksheets to check the children's progress and provide practice are helpful, especially when machine time per child is limited.

COMPUTER PRACTICE
30 OCT. 78
NAME ______________________

Simulate these computer runs. Show your "printout" on the screen.

```
10 CLS
20 PRINT "BIG"
30 PRINT
40 PRINT "YELLOW"
40 PRINT "BLUE"
60 END
```

```
10 CLS
20 PRINT "THE ANSWER"
30 PRINT 30*2
40 PRINT 30 + 2
50 PRINT "30-2"
60 PRINT "THE END"
70 END
```

Here is a program and "printout." Find and fix the mistakes in the programs so a run will produce what is shown on the screen.

```
10 CLS
20 PRINT 20 + 6
30 PRINT 30 + 4
35 PRINT
40 PRINT "60-3"
50 PRINT 10-10
60 PRINT "HELLO"
70 END
```

```
26
34
57
0       HELLO
```

To build the complexity of their programs, the children now need to learn the concept of a variable, and GOTO statements. Trying to teach variables to children who have had no algebra frightens many teachers unnecessarily. The simple picture of a series of mailboxes, all labeled with a name or a letter, and holding different

Kids, con't...

numbers as specified, works beautifully.

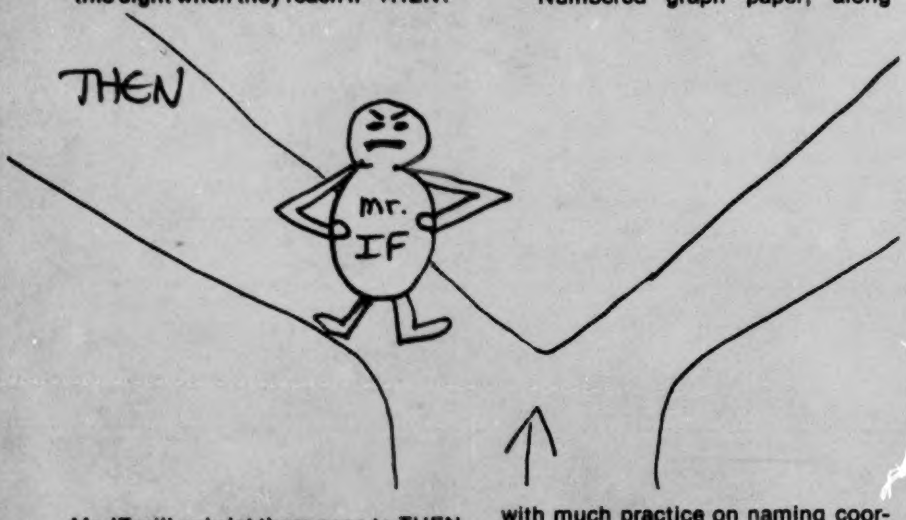
A	B	C
5	3	7

If the "mailboxes" are drawn on the chalkboard, a simulation of a program can be traced carefully, and most children will have no trouble understanding how a memory works. In this case, they have an advantage over the algebra student, in that eight year olds see nothing peculiar about the statement

$$X = X + 1$$

With the addition of INPUT and RND functions, the student now can produce quite a wide range of programs. To complete a beginner's course in text programs, the more difficult IF-THEN and FOR-NEXT are taught in a very concrete fashion.

The children picture themselves as "traveling" through their own program, doing each of the statements in turn, and imagining they see this sight when they reach IF-THEN:



Mr. IF will only let them pass to THEN if they meet his test. Otherwise they must proceed down the open branch of the path. When deciding on the test to go on to THEN, they must figure out who Mr. IF wants to go down his path, or who he wishes to exclude.

FOR-NEXT statements are best saved until last, and shown as a shortcut method for accomplishing a more complex list of simpler statements.

```
 5 X=1
10 PRINT X
15 IF X=4
   THEN GOTO 30
20 X=X+1
25 GOTO 10
30 END
```

```
 5 FOR X=1 TO 4
10 PRINT X
15 NEXT X
20 END
```

The basic tools of the graphics program, SET and RESET, along with the other function statements covered earlier, enable the children to make their own initials on the screen

(always a big favorite), draw pictures of objects, and design a myriad of programs, using horizontal and vertical lines, as well as individual points.

Numbered graph paper, along with much practice on naming coordinates of a point, contribute to the success of these programs. On a system with color graphics, such as the Apple, beginning with graphics programs, rather than starting with PRINT statements, is a natural. But either approach works well.

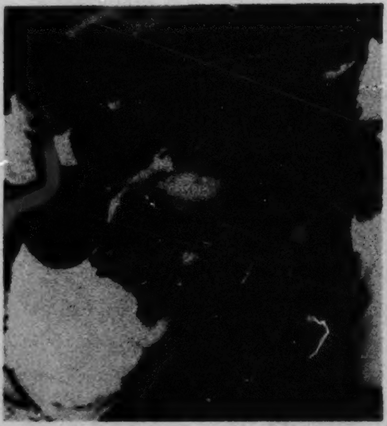
Once the children are able to write a program without consulting their notes for statement meanings, and are able to conceptualize a program from beginning to end without the use of written flowcharts, they are ready to evaluate their work under the headings of **efficiency** and **creativity**. The teacher must stress that there are many ways to write the same program, just as there are many ways to express the same idea in English, but some of the ways are awkward, or have extraneous steps, or could be approached better from a different view of the problem. As can be imagined, it takes a great deal of work before an elementary school programmer reaches this point, and some never will until they become older.

The last portion of the teaching framework, looking at new uses for computers, can vary from collecting information on existing systems and research, to inventing new systems of their own. This is a natural place to discuss hardware and software, and form opinions on whether computers can "think."

Soon after the children are able to write their own programs, prepare yourself for the following events:

1. If you want to use your school's computer, you'll need to make a reservation a week in advance.
2. Parents will call you and want to know why their child is suddenly speaking a foreign language, with words like "do-loop" and "glitch."
3. Santa Claus will be having a few choice words with you.

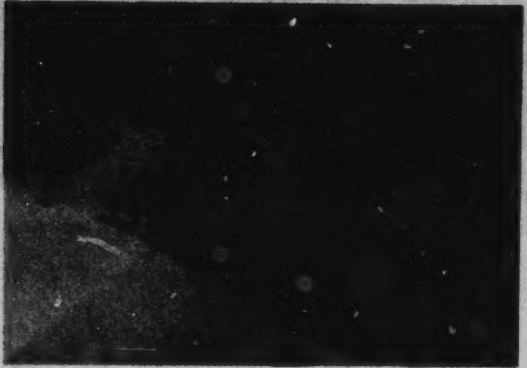
I have found computer programming to be an exciting way of teaching thinking skills, mathematics, and problem solving. It is highly motivating for children whose abilities range from average to very bright, who have enough reading and number skills to operate the keyboard. It gives young children a view into their future, while at the same time seeing the present in a new light.



I am anxiously awaiting the day when these kids are college freshmen, and they walk into their computer science course with **ten years** of programming already under their belts at age eighteen. The implications for their futures and ours stagger the imagination. ☐

---

*from "The Apple Corps: An Introduction to the Apple II for Children," by Sally Greenwood and Dr. Donald Piele, 1978.



Micros "GOTO" School

Donald T. Piele

Microcomputers can be used in the classroom for: instructional activities that we associate with CAI (Computer Assisted Instruction) or CMI (Computer Managed Instruction); enrichment activities that we associate with simulations and games; making numerical calculations for the purpose of solving mathematical problems; teaching students how to program - primarily in the BASIC language; and individual exploration of original problem-solving.

Each lesson consists of a simple program with a short explanation of the new statements, a sample run, and a series of simple program changes for the student to do.

This article is focused on the latter activity. It is a report of a pilot project in which a microcomputer was placed in a sixth grade classroom for 8 weeks for the purpose of developing logical thinking skills. The students were given instruction on how to program the APPLE II microcomputer to draw color graphics designs. They were then given similar problems to solve using the commands they had learned.

An Apple For The Teacher

In the Spring of 1978, I contacted Gordon Kunaschk, a sixth grade teacher at Bose Elementary School in Kenosha, Wisconsin. He was receptive to the idea of giving up two hours a week of class time for eight weeks to let me teach his sixth graders how to program a microcomputer. If nothing else, it would be a lesson in computer literacy. Gordie had never programmed a computer before, but he was willing to learn along with the kids if I was willing to provide a computer and the necessary instruction.

The Center For The Application of Computers at UW-Parkside supported the idea and supplied an APPLE II microcomputer for the project. This was a fortunate choice for us since the APPLE II system is easy to use: it is portable; it has a good keyboard; and most important of all, it has a very simple and natural set of graphics commands that allow the programmer to create pictures on a TV screen using 16 different colors. The ideas I wanted to emphasize about computer programming would be considerably enhanced by a graphics display. The basic programming construct of a loop, for example, could be visualized, and every problem to be solved by the students could be represented by a single picture.

Getting Started

My objective for bringing a microcomputer into a sixth grade classroom was to create an environment for **active problem-solving**. The BASIC programming language statements, enhanced by the graphics of the APPLE II microcomputer, form the logical building blocks. Each lesson consists of a simple program with a short explanation of the new statements, a sample run, and a series of simple program changes for the student to do. These activities allow the student to discover how the statements in the program effect its outcome. Also, problems are posed that require the student to combine statements in sequential order to solve a problem. As a result of working on these questions, the student gets a working understanding of practical problem-solving skills such as:

1. Understand the problem, its givens and goals
2. Make conjectures and probe the problem by trial and error
3. Decide on a set of possible methods of attack
4. Evaluate each possible approach for its correctness
5. Reflect on successful solutions and generalize

Each of the following exercises was designed to provide practice for these skills. They are samples taken from a larger collection and are not contiguous lessons.

Lesson #1

Key Words: GR, COLOR, PLOT

LIST	EXPLANATION
10 GR	The computer is put in GRaphics mode.
20 COLOR = 9	
20 COLOR = &	The COLOR is set to orange. There are 16 different colors to choose from.
30 PLOT 10,15	The position 10 over, 15 down from the upper left hand corner is plotted.
100 END	

RUN

Programming the microcomputer was considered by the sixth graders to be highly motivating. They would rather spend their recess on it than go outdoors.

Your Turn (RUN the program after each change)

1. Change line 20 to . . 20 COLOR = 3
2. Change line 30 to . . . 30 PLOT 5,7
3. Add line 50 . . . . 50 PLOT 5,5
4. Add line 40 . . . . 40 COLOR = 6
5. Delete line 40 40
6. Add a point that connects 5,7 with 5,5 . . . . 60 PLOT ?,?

Don Piele, University of Wisconsin-Parkside, Kenosha, WI 53141.

From *Creative Computing,* Volume 5, Number 9, pages 132–134. Copyright © 1979 by Creative Computing. All rights reserved. Reprinted with permission.

School, con't...

7. Add line 30 ...... 30 PLOT 39,39
8. Add line 40 ...... 40 PLOT 40,40
9. Delete line 40 40
10. Write a program that will display the first letter of your first name in graphics.

Lesson #5

Key Words: FOR-NEXT

LIST	EXPLANATION
10 GR 20 FOR I = o TO 15	Begin a loop with I = 0 and increase I by one each time until I = 15.
30 COLOR = I	The color changes with each pass through the loop.
40 PLOT I,15	The position to be plotted changes with each pass.
50 NEXT I	End of the loop. Go back to statement 20 if I is less than 15. Otherwise go to line 60.
100 END	
RUN	

Your Turn (RUN the program after each change.)

1. Change line 40 to .... 40 PLOT I,10
2. Change line 40 to .... 40 PLOT 20,I
3. Change line 20 to .... 20 FOR I = 0 TO 39
4. Change line 40 to ..... 40 PLOT I,I
5. Add line 45 .... 45 PLOT 39-I,I
6. Change the program to draw + in graphics.

Lesson #10

Key Words: RND, IF-THEN

LIST	EXPLANATION
10 GR 20 COLOR = 9 30 X = RND (40)	A random number is chosen from the numbers 0 to 39 and put in X.
40 Y = RND (40)	Another random number in chosen and placed in Y.
50 IF Y>20 THEN COLOR = 3	If Y is larger than 20 then change the color to blue(3).
60 PLOT X,Y 70 GOTO 20 100 END	Go to line 20 and repeat.
RUN	

Your Turn (RUN the program after each change.)

1. Change 50 to .... 50 IF Y > 10 THEN COLOR = 3
2. Change 70 to .... 70 GOTO 30
3. Change 70 to .... 70 GOTO 20
4. Change 50 to .... 50 IF X > 20 THEN COLOR = 3
5. Add 55......... 55 IF Y > 20 THEN COLOR = 13
6. Delete 55 55
7. Change 50 .... 50 IF X + Y > 40 THEN COLOR = 3
8. Adjust the program to plot 4 different colors in the four different corners of the screen.

Tell & Run

In addition to the lessons, the students were given short problems to solve. They were asked to predict the output of a given program before they observed it run on the TV screen. This gave the students a chance to test their ability to reason sequentially through the statements of a program. A few examples are given below.

#1	#2
10 GR	10 GR
20 FOR I = 0 TO 10	20 FOR I = 10 to 20
30 COLOR = 9	30 COLOR = I
40 PLOT 2,I	40 HLIN 0,39 AT I
50 NEXT I	50 NEXT I
60 END	60 END

The original purpose for doubling up was to provide more computer time for the class each week. But it turned out to be valuable for a completely different reason - cooperation.

Other activities reversed the process and presented a picture and asked the student to write a program that would produce the same result. Here are a few examples:

Computer As A Creative Tool

The APPLE II microcomputer was left in the classroom during the week to give the class time to experiment. Students signed up in pairs to work on the exercises together. The original purpose for doubling up was to provide more computer time for the class each week. But it turned out to be valuable for a completely different reason - cooperation. The students helped each other figure out the effect of each new command. The programming exercises facilitated discussions about the behavior of each new statement. New discoveries were shared with pride and enthusiasm. The computer was the focus and facilitator for cooperative problem-solving.

Student Reactions

After eight weeks, the students were asked to respond to the following questionnaire, using a scale of 1 to 5 (1-strongly disagree, 5-strongly agree), with responses from the 6th grade class of 14 boys and 10 girls recorded.

Reflections

Only a small sample of the exercises done by the students are presented in this article. An entire collection of problems was prepared for the 6th grade class to be used on the APPLE II. At the present time, good materials are not readily available. This presents a formidable obstacle to the inexperienced teacher who wants to use computers in the classroom. As more classrooms begin using microcomputers and sharing their work with others, this problem will diminish.

Students in this sixth grade class were very enthusiastic about working with a microcomputer. In contrast, students at a nearby high school who had not been exposed to computers before were generally uninterested in learning how to use them. Perhaps by this time, the older students have other activities that are more relevant. Also, in the sixth grade the survey shows that boys and girls are equally confident and interested in programming the computer. However, in entries from 10th graders in an annual computer problem-solving contest held at UW-Parkside, the boys outnumber the girls 9 to 1. A recent survey in Creative Computing Magazine had a response with a distribution of 95.4% male and 4.2% female.

Programming th[illegible]crocomputer was considered b[illegible]h graders to be highly moti[illegible]ey would rather spend their recess on [illegible]t than go outdoors. Students came early to school and would hang around as long as they could after school. A sign-up sheet became a necessity. With practice, some of the students became resident 'experts' able and thrilled to help others - including the teacher. Some of the sixth graders entered our annual computer programming contest.

Research Questions

This pilot project suggests a number of possible topics for further investigation and research:

1. If students learn how to program a computer early, will they maintain their enthusiasm in later years?
2. If students learn how to program a computer early, will the interest and confidence level of girls, in later years, continue to match that of boys?
3. What factors influence the acceptance of microcomputers in the classroom? Graphics? Games? Programming problems?
4. What is the relationship between logical thinking skills and creative programming skill?

Conclusion

Instead of the computer programming the student, the student learns how to program the computer.

Computers have been used in education primarily as a **delivery system** for subject matter. This role will continue to be developed even further with microcomputers. However, a new application is emerging which is fundamentally different. Instead of the computer programming the student, the student learns how to program the computer. Arthur Luehmann describes it as follows:

> "Computing constitutes a new and fundamental intellectual resource. To use that resource as a mere delivery system for instruction, but not to give a student instruction in how he/she might use the resource, has been the chief failure of the CAI effort. What a loss of opportunity if the skill of computing were to be harnessed for the purpose of turning out masses of students who are unable to use computing."

The computer as an instrument for learning logical thinking and problem-solving skills is only beginning to be understood. However, with the rapid development of low cost microcomputers in the next few years, computers - and hence computer problem-solving techniques - will become a fundamental intellectual resource. □

The Hampton City Schools Computer Program

NEDRA I. HARKAVY

Philosophers have long maintained that diversity is essential to a well-rounded, productive life, and in no area of life is this need more urgent than in education. A mind, given the proper amount of freedom and discipline, will expand to accept new ideas. This truth, so well known yet so often overlooked, is revolutionizing contemporary concepts of education.

Foremost among educational tools is the computer. Around 1971, Hampton City Schools decided that learning to use computers would be a valuable skill for gifted students. The district tied into a computer at the College of William and Mary, but the 35-mile distance meant expensive long-distance calls, and the district could not set its own priorities for the rented time. With a 3-year Title IV-C grant of $350,000, the district rented a large computer for administrative purposes and for the gifted program. Although the grant money has long since run out, the district will own the computer as of 1980 and defray costs by renting time to other systems.

Gifted children are often called "ivory-tower kids." They like to work alone and they love to work with a computer. The younger they start, the easier it is to learn—almost like learning another language. Students have been trained in the operation of computer terminals and use this knowledge to aid them in such varied subjects as reading, mathematics, and the sciences. Computer-aided instruction has proved very successful as well as enjoyable to the students, who spend hours on the computer absorbed in what it has to offer. The district is beginning to use the computer for remedial work. It's a fantastic tool for both ends of the spectrum, as it gives the child immediate reinforcement.

By providing diversity, the computer gives the students the opportunities and the motivation for self-expression. For a creative writing program on haiku, the computer presents the learner with nouns and adjectives with which to create a poem. A science course programmed with local data enables students to determine effects of pollution on the James River. A program on the stock market, which simulates financial operations, encourages the student to use deductive reasoning to solve problems. The programs are updated yearly, some monthly. Computer instruction is less expensive per child than textbooks. Gifted children are not happy learning from obsolete texts. An additional turn-on is that it's so current.

Gifted high school students learn programming and then design programs for elementary pupils. By second or third grade, the children have learned computer literacy; by fifth or sixth grade they are doing their own programming on a simple level. Students type into the computer, which can respond by printout or video tube. It speaks their language. It tells them "Right on!" "Groovy!" "You've done it!" When the kids use four-letter words, it tells them "No, no." It "talks" directly to the child by name. The computer also helps gifted students explore careers by responding to their interests with descriptions of vocations, kinds of work involved, and training necessary to enter the field. Jobs available locally are listed on the computer.

The Hampton City Schools computer program provides an ongoing opportunity for students to supplement and complement their regular educational curriculum. Knowledge about computers has been the entrance to jobs for many students, including summer work during college. With everything so computerized now, it has helped them in their studies of law, medicine, and business. They have never been sorry they learned it. This is the age of technology. The future is here. The computer is a "must" tool, especially for the gifted and talented.

For more information on the Hampton City Schools program contact: Nedra I. Harkavy, Director, Program for the Gifted, Hampton City Schools, 1306 Thomas Street, Hampton, Virginia 23669.

The Paducah Tilghman High School "Chemics" Program

STEVE JOHNSTON

The "Chemics" program at Paducah Tilghman High School was started in 1977 to satisfy the need in our curriculum for a course to challenge and stimulate gifted/talented math and science students. Our school system had previously initiated a combined studies course, American Studies, for gifted/talented high school juniors. This course consisted of a 2-hour block of English and U.S. History. We also had a math-science-English enrichment program in the elementary schools. The need for more challenging courses at the high school level was recognized and investigated. After much study, we found that what we wanted was a combined studies course for seniors with an emphasis on math and science.

The Chemics Curriculum

The results of our efforts was a restructuring of three courses that were already in the curriculum: second year Chemistry, first year Physics, and first year Computer Science. The students soon coined the name "Chemics" for the course. We used a 2-hour block of time; however, instead of team teaching, the class of 30 students is divided into two groups. We use flexible scheduling with each group, with three 2-hour labs and four 1-hour lectures per week, and at least one lecture with both groups together. We found we could get 3 to 4 hours of work done in the 2-hour labs because of the time saved putting up and taking down experiments. Schedules normally run on a 2-week sequence, but can be altered to conform to individual course needs or school disruptions and holidays.

Criteria for student participation is based primarily on teacher recommendation. Since the course includes second year Chemistry, we rely heavily on the recommendations of the first year Chemistry teachers. The course has been very popular and we have had no problem filling the class with qualified students. One reason is that the type of students we want are very academically oriented and usually have full schedules. Being able to offer them 2½ credits in 2 class hours (one each for Chemistry and Physics and ½ for Computer Science), has been a real advantage. One problem with the 2-hour format is that great care must be taken in placement of the 2-hour block in the master schedule. Since the classes these students take, such as Calculus, are not usually offered more than 1 hour a day, care must be taken to avoid conflicts.

Although we do not team teach, we do try to see that course content is coordinated to integrate all three subjects. We have been particularly successful with computer-generated dry lab simulations in the Chemistry and Physics courses.

Choosing a Microcomputer

After the first year, we found that the opportunity for lab simulation with a computer necessitated changing the method used to teach Computer Science. For the previous 6 years, we had two keypunch machines in the classroom and batch-processed our computer programs at Murray State University. This arrangement had some advantages and disadvantages. We were able to teach FORTRAN, had hard copy for output, and the students were exposed to programming under a punched card environment. The drawback was a turnaround time of 2 to 3 days.

After researching timesharing, minicomputers, and microprocessors, we decided that the best solution for our situation was one of the microprocessor-based computers. After looking at all of the available brands, we decided to purchase the Apple II. Even though it was more expensive, the color graphics and the expandability seemed to best suit our needs. The purpose of this article is not to sell a particular brand of computer; but if you are in the process of buying one, try not to let cost be the primary factor—this is one educational area in which you really do get what you pay for. We had to spread our purchase of equipment over several years and were not able to get the computer, disc drive, and printer at the same time. It is extremely difficult to teach Computer Science with programs stored on cassette tape and without access to hard copy. So if at all possible, get the disc drive and printer with the computer.

We are extremely pleased with our Apple and have continued to add to our equipment each year. Next year we will have a total of five Apples with two disc drives each, two printers, a graphics table, and the ability to teach four programming languages: BASIC, FORTRAN, PILOT, and PASCAL.

The Lack of Educational Software

The one discouraging area in the conversion to the microcomputer has been the lack of educational software and course material. The manufacturers have done a superb job in developing the machines, but the development of usable educational application materials is far behind. This situation is apparently improving; you do see more and more advertisements for educational software, which is great except for the fact that the number of disreputable companies will also increase. Almost all of the programs being sold are really quite good, but we have bought a few that were worthless. If at all possible, know what you are buying and who you are buying it from. Also, do not be afraid to write your own software. You will be surprised at what you and your students can turn out. After the students acquire the tools necessary for programming, they can find challenge in the area of application software. Applying computer programming to what they are learning in Physics and Chemistry has been an important factor in motivating the students in all three areas.

Information Resources

If you are just starting to implement a program involving a computer, or think you might be interested in this area, there are several things you can do to broaden your background in this area. First of all, *read*. There are many good monthly magazines available that specialize in the microcomputer. The problem is that most of these are not available in newsstands. The best source for these publications is the computer store. If you are fortunate enough to have one in your area, don't be afraid to talk to the salespersons about your particular application. I have found most of these people knowledgeable and interested in all aspects of computer applications. While you are there, ask to try out one of the machines. Most of the manufacturers market a computer-assisted instruction program to acquaint the user with the machine and the programming language.

Another resource is the local business community. Ask your students if any of their parents work in a business that has a microcomputer. You will be surprised at the number of small businesses that are using computers, and most of these people love to show off their hardware and share their ideas about computer literacy.

Computers are not a cure-all for the ills of education, but they are a valuable tool that can and should be used, not only to teach about computers themselves, but also to enrich instruction and challenge students in other subject areas.

For more information about the Chemics program contact: Steve Johnston, Paducah Tighlman High School, Paducah, Kentucky 42001.

Computing at a New Public High School for Gifted Students

Photo by Thad Sparks

From *onComputing*, Volume 2, Number 4, pages 72–75. Copyright © 1981 by onComputing, Inc. All rights reserved. Reprinted with permission.

We believe the responsibility of challenging talented students and preparing them for tomorrow's world requires commitment and flexibility. Computing will be a part of that future world and plays a large role in the education of the 150 North Carolina School of Science and Mathematics (NCSSM) students during their first year, a role which will increase as the school grows. The successful integration of computers and education will require a commitment of funds, support by the faculty, and a well-defined philosophy toward the uses of the computer.

We have made our first steps in each of these areas. It is difficult to adequately express the excitement of teaching talented students at a time when microcomputers are within the reach of most schools, in a location providing access to an outstanding educational computing network, and where the support of private industry will enable the school to purchase a minicomputer with a 32-bit central processing unit (CPU). This article is a description of our enthusiastic steps to implement this program.

We use the computer, a particularly effective tool in educating gifted students, in three important ways in the education of our high school juniors. Our most important use is in the classroom. We plan to involve the computer in the classroom instruction of subjects such as American Studies, foreign languages, and music, as well as the traditional uses in mathematics and science. This use is more characteristic of the university than the high school.

The second principal use is in the area of independent study. Since motivation is essential for successful independent study, the natural interest computers arouse in students is an immediate asset. Independent study was once viewed as useful only for the exceptional student; however, with the aid of the computer, many projects can be formulated and successfully completed by students from a broader ability range. We intend to provide a high proportion of our student body with the satisfaction coming from independent study.

Our third major use is as a topic of study in its own right. Such subjects as programming languages, algorithms, business and scientific applications, and the impact of computers on society are offered. While these topics are treated in other classes and in independent projects, each student has an opportunity to take a course which addresses these broad areas in a formal manner.

Photo by Joe Liles

Integrating the computer into the total school program is a priority at NCSSM. Each graduate of the two-year program will have used the computer as a tool, will have an appreciation for the uses and limitations of computers, and will have written several programs. Implementing this philosophy requires the cooperation of the entire faculty and the availability of excellent computing facilities. Our first faculty is establishing a tradition of a broad view toward teaching and education. This approach—team teaching, cooperation between teachers of different disciplines, and the feeling of a team working toward the same goal—creates an atmosphere where teachers encourage students to make use of technological advances and which encourages the testing of new approaches or the inclusion of new topics.

The computing facilities consist of microcomputers and terminals accessing a minicomputer or large mainframe. We have five Apple IIs, one TRS-80, and four terminals connected to the Triangle Universities Computation Center (HP-2000, Amdahl V-8, IBM 370) through North Carolina Educational Computing Service (NCECS). Next year we will add two more Apples and purchase a Digital Equipment Corporation VAX-11/750. These facilities, together with our membership in NCECS, will give us a combination providing students with access to a wide range of computing power and virtually an unlimited number of potential projects.

The planned acquisition of the new VAX is made possible by a gift from a private corporation. This is one of several gifts from private corporations which represents a remarkable partnership between the public and private sectors. The VAX provides an excellent base for the future growth of the school. In our opinion, it offers effective interactive power. We believe there is no substitute for an interactive environment when working with young students. The bulk of our computing on the VAX will be in Pascal and C.

The microcomputer is the centerpiece of this year's program and will be the major part of our program for the next several years. Microcomputers such as the Apple II and TRS-80 are an excellent source of graphics and BASIC. The graphics capability of the Apple provides opportunities which we will pursue in the living/learning atmosphere at NCSSM. The residential aspect of our program enables us to make efficient use of our facilities—the computer room is open from 8 AM to 10 PM. Teachers can give an assignment involving the use of a microcomputer to augment classroom instruction, knowing the necessary facilities are available to the student.

We believe microcomputers can be used effectively to augment course work, especially in mathematics and science. Students are willing to spend more time "probing" into a subject when it is taught in conjunction with an outside-of-class interest. The graphics capability of a microcomputer interests most students and can bring life to a textbook topic.

The Apple Education Foundation has given NCSSM the equipment to develop software for implementing our ideas. Initially, we will focus our efforts on the coordinated use of the Apple in mathematics, biology, chemistry, and physics. The mathematics teachers will discuss the numerical approximation of experimental data, run simulations, and allow students to explore the topic interactively (making extensive use of graphics). The science teachers will then build on this experience. The Apple will be used to process data from an experiment performed by the students, an exciting use of the microcomputer.

The research laboratory of today uses machines both to acquire and to process experimental data. Dedicated equipment is too expensive for consideration by a high school, but the flexibility (and cost-effectiveness) of microcomputers can help bring the laboratory experience of high school students closer to the methods used in the actual research labs of today. Essentially, the microcomputer can be used to demonstrate the "spirit" of the automatic acquisition and processing of experimental data.

The computer is also useful as a bridge between subject areas. It is important for schools to take a broad view toward the education of eleventh and twelfth graders. This is especially true for a school of science and mathematics. Specialization is important and a reality, but it is also important that students receive solid exposure to a variety of disciplines and that the relationships of the disciplines be emphasized. The computer, with its applications in many areas, can provide not only examples of faculty cooperation across disciplines, but also opportunities to further knowledge in one area by applying knowledge and technology from another area.

NCSSM expects to serve several constituencies. Three of these constituencies—the students at NCSSM, the students at other schools, and the teachers throughout the state—exert a major influence on the direction computing will take at our school. Our emphasis on microcomputers and graphics is based in part on the availability of the equipment for schools across the state. We are committed to sharing the results of our experiences with educators throughout the state for our mutual benefit. This commitment will be translated into designing materials which are applicable statewide, and into presenting summer programs for teachers from across the state. For our first summer program for teachers, we are planning two workshops on the uses of microcomputers. One hundred and twenty teachers will be invited to spend two weeks in residence to learn elementary computer programming and the applications of microcomputers in mathematics and science.

We believe the next two years are critical in involving microcomputers in secondary education. If the use of this new technology can be given direction and coherence, microcomputers will become an integral part of classroom education. We believe this is desirable and hope to demonstrate that microcomputers can be used for far more than the teaching of computer programming. The integration of microcomputers into the mathematics and science curriculum will be a major focus of the next several years at NCSSM.

The inspiration and support of a farsighted governor, the people of North Carolina, and the hard work of the founding faculty and administration have combined to create a learning environment at NCSSM that is ideal for developing new approaches and for helping other educators in their efforts to improve education at the secondary level. In the course of designing a curriculum for talented students, we hope to develop materials and programs appropriate for use in many schools across the state. The partnership of government and private industry is enabling us to meet these important goals.■

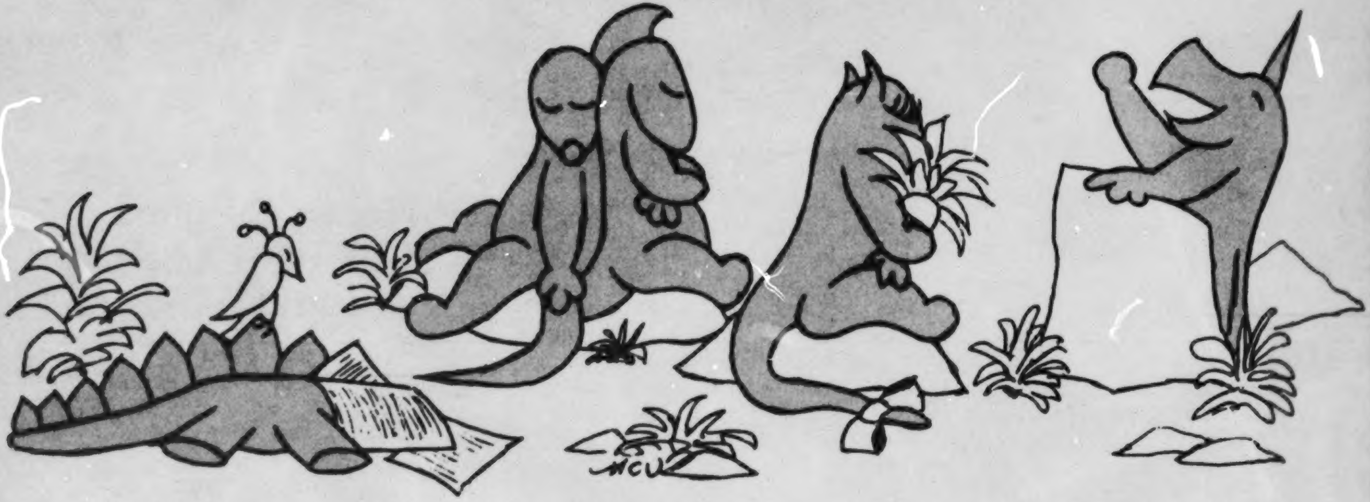
North Carolina School of Science and Mathematics

The North Carolina School of Science and Mathematics was created in 1978 by the North Carolina General Assembly at the suggestion of Governor James B Hunt, Jr. It is a statewide, residential, coeducational, public high school developed specifically to provide a challenging education for eleventh and twelfth graders who are gifted in science and mathematics. The school is an important step toward reversing devastating national trends—the neglect of education for high-caliber students, the decline in the quality of science and mathematics education, and reduced economic productivity.

The school opened with its first class of 150 eleventh graders in September of 1980. Enrollment is planned to increase by 150 each year to a maximum of 900 eleventh and twelfth graders. Eventually, up to 15% of the student body will come from out of the state. The school is located on the former Watts Hospital and School of Nursing complex in Durham, a gift of 27 acres and fifteen buildings from Durham County. The student population is 50% male, 50% female; 24% are from minority groups. The students come from both rural and urban areas, and represent 63 of North Carolina's 100 counties.

The school has attracted strong support from foundations and corporations. The reality of the school's growth and its future success will be a direct result of an extraordinary partnership of the public and private sectors.

Computers . . . . Are All Dinosaurs Dead?



Are all dinosaurs dead? — Probably not, and the six to eight year olds studying this subject in the University of South Alabama's Saturday Program for Gifted children are using the Program's TRS-80 microcomputers to gain background knowledge in the subject. Not only are the youngsters learning more about one of their favorite topics through the use of an instructional game but also, as a byproduct, they are learning to operate the computer, an item of equipment that is being increasingly used in programs for G/C/T youth.

Numerous articles have been written to describe ongoing computer oriented programs and their successes, yet none has provided the "How To" information for an interested educator to initiate and manage such programs in his/her own environment. In keeping with G/C/T's intent to provide usable information to those involved with the G/C/T child, this article will be directed toward informing the reader how microcomputers can be reasonably introduced into an existing program for G/C/T children.

One might easily relate the traditional large scale computer in the pre-college educational enviornment to the dinosaur: its dominance in the educational realm is on the wane. The reason for this situation is the development of the microprocessor—a computer on a single integrated circuit. This device, as the heart of a microcomputer, offers computing power equivalent to computers costing in the five to six digit range only a few years ago.

Even those schools which are using microcomputers or the larger minicomputers seldom seem to exercise these equipments effectively. In many programs one finds the computer used for a relatively few students to learn programming techniques or computer design. The computer may be infrequently used by an additional segment of the student population for simulations or games that are related to political science, history of business. In other schools one finds the computer used exclusively as a "teaching machine", most frequently for teaching younger children reading, math or perhaps a foreign language. The computer is a flexible, dynamic teaching aid which should be utilized across the entire spectrum of its capability. It is time to pause and take an arm's length look at the computer and its role in the school environment.

The University of South Alabama Saturday Program for Gifted Children (SGP) began offering computer programming for its students in the spring of 1977 using the University's IBM Model 370/145. This was at least a step in the right direction. However, the complexity of dealing with such a large system and its uncompromising protocol made it less than optimal, particularly for the younger children (8-9 year olds) in the class. Even this first foray into computing verified the high level of interest among the SGP students. Approximately seventy percent of those who had participated in the Spring programming course and later enrolled for a second year of the Saturday Program, also chose a second course in computer programming as one of their Fall Quarter courses. Still, the complexity of working with the IBM system caused some consternation.

In the Fall of 1977, The Tandy Corporation's Radio Shack Division announced the availability of its TRS-80 microcomputer. This self-contained, completely assembled microcomputer appeared to be the exact answer to our needs. Using the state-of-the-art Z-80 microprocessor, this system is composed of a computer/keyboard, video monitor and cassette recorder, at a total cost of $599.00. The Saturday Program had been looking for a microcomputer to use but several problems had precluded the acquisition of systems

From G/C/T, September/October, 1978, Number 4, pages 16–17, 46–50. Reproduced with permission of G/C/T, P.O. Box 66654, Mobile, AL 36660.

prior to the entry of the TRS-80. First, most reasonably priced (under $1800) microcomputer systems, that were available in the Fall of 1977, were sold in kit form. This required the buyer to assemble and test the system prior to putting it into service. Due to the electronics background and sophisticated test equipment required to accomplish this task, it was an insurmountable obstacle for most schools. Second, repair and maintenance service for the existing systems was either unavailable or prohibitively expensive. With Radio Shack's nationwide system of repair facilities, these roadblocks were overcome. Tandy has established a separate division, Tandy Computers, which will most probably eventually direct the corporation's computer related activities; however, the TRS-80 is now a standard "off the shelf" item at most Radio Shack stores.

In November, The Saturday Program ordered four TRS-80's. They were no more difficult to set up and operate than a simple component stereo system. Plugging in three A.C. line plugs and three interconnecting cables is all that is required to make the system operational. All connections are clearly marked. The manual provided with the microcomputer assumes that the user has no prior association with computers and thus takes the reader from initial equipment setup through all the programming techniques available in the computer's resident BASIC Language. The manual is written in a step-by-step, programmed instruction approach which will permit its use either as a resource book for the teacher or as a student text.

The primary use of the SGP microcomputers is to teach computer programming to students in the age range of 8 to 18 who elect this subject. During their first quarter of programming, the students are taught the computer language BASIC. The objective is to familiarize the student with the computer and how it can be made to work for him/her. Some simulations and computer games are used as motivators during the quarter. It became apparent that many of the kids would take this course repeatedly for the simulations/games alone, without ever mastering BASIC. To ensure their concentration on the task of learning to program, a post-test has been introduced on which a score of 90% must be achieved in order for the student to select a subsequent programming course. If a program intends to offer computer time to students beyond an introductory level, the participants must be guided and challenged just as with any other course for G/C/T youth. The SGP present modus operandi is to have the advanced programming students identify one or more objectives which they will work toward during the quarter. These objectives might concentrate on improving their skills in graphics techniques or in the development of programs for home use. The students work independently on their projects, using their instructor as a resource person for advice or/and assistance as required.

Using the independent study approach to advanced programming students can reap other benefits for both the program and the student. The student may elect to develop one or more programs relevant to his activities in another of his subjects. Such a student could be of valuable assistance to a teacher who wishes to include some computer related activities in his/her course.

Beyond the student directed independent study, these advanced programmers could be encouraged to do specific developmental tasks related to the administration, planning or curriculum of the program itself. There are an almost unlimited number of software items that could be developed by these programmers which might make a significant contribution to the direction in which the program grows.

The student programmers' growth and development are simply functions of the teacher's own creativity in keeping them challenged. If the teacher follows the guildelines described above, she/he can keep these more capable students deeply involved in their own research and development activities while the majority of classroom time could be devoted to beginning programmers or those who are having difficulty mastering programming concepts.

Now, what about students who are not in programming courses? Teachers should consider using the microcomputer to add spice and motivation in their courses in much the same way as other audio-visual equipment can be used. Beyond the rudimentary machine functions which any other computer user must learn (it takes approximately five minutes to teach anyone), the computer does the rest of the work. Programs should be written in a fully interactive mode (i.e., self-prompting and error correcting) so that the time wasted by instructions from the teacher will be minimized. An example of such a program accompanies this article. This program, "Dinosaurs", is written in Radio Shack Level I BASIC and was developed for use with a course for six to eight year olds entitled, "Are All Dinosaurs Dead?" This type of program is easily developed and can be expanded or reduced depending upon the capabilities of the microcomputer available. The example program requires the TRS-80 with 16K of user memory. With very little effort the program could be reduced in scope to fit the smaller TRS-80 with 4 K of user memory. (The reader should not be frightened off because of a little technical jargon such as 4K or 16K of user memory. Within an hour, she/he will find the terms coming quite naturally.)

What are the hard facts an interested G/C/T teacher must have to sell his/her administration on such a project? How should the program be implemented and how much will it cost? Many G/C/T programs which offer courses in computer programming might more appropriately call these courses "Introduction to Computers" because the kids get little or no opportunity to have hands-on experience with the computer. *G/C/T kids have little interest in watching others operate equipment!* They must be actively involved. The Saturday Program has found that the most desirable ratio of students to the computer input/output (I/O) device is two to one for classes on programming. The I/O device might be a printer or video terminal in a large computer time-sharing system, while the microcomputer such as the TRS-80 has its keyboard and video display integral to the microcomputer system. The rationale for the two to one ratio has evolved over several quarters of operation. Due to the usual time constraint of one hour per class, three or more students per I/O device will not allow each student to have sufficient hands-on opportunity to work with the programming techniques being taught. Hands-on activity is the key to continued interest and achievement in this subject area. The SGP violated its own rule in this regard one quarter with disasterous results. Because of the large number of applicants for programming, the classes were set up with a three to one ratio with the result that students who participated in these larger classes did not have sufficient applications time to learn the programming techniques. Many of them became disenchanted with the whole idea because of their competition to get a share of computer time. Those who were more persevering made the best of the unfortunate situation and they signed up for a second opportunity to learn the material during the following quarter. Needless to say the ratio of students to I/O device is back to the proven two to one.

One might logically ask, "if two to one is good, why wouldn't one to one be even better?" Ironically enough, even if there is an I/O device for each student in the Beginner's Programming Classes, during the initial learning/investigative stage, the students will gravitate toward working in pairs to solve programming tasks. Learning a computer language is much like learning any foreign language. The combination of learning the language and applying the language to problem solving situations often causes even the most dedicated "lone wolf" to seek association with his peers to accomplish a task. One might view this interactive environment as one of the major advantages to teaching computer programming in this manner.

The number of systems a G/C/T program is to acquire will be a function of the educator's creativeness in locating funds and/or his/her persuasiveness with the local school administration. If there is absolutely no computer equipment on site, an initial investment in one or two systems could get things started. An individual adept at proposal writing certainly should seek outside resources and try for a larger number of systems.

An interesting cost comparison can be shown between the relative merits of adding printers or video terminals to existing large computers available to a school system or beginning from "scratch" by purchasing an independent microcomputer. Consider the cost of two of the more common I/O, terminal equipments:

Lear Sigler, ADM-3A, video terminal	$895.00
or	
Digital Equipment Corp DEC Writer II, printer terminal	$1495.00
vs.	
Radio Shack TRS-80, microcomputer	$599.00

The terminal cost only includes the equipment (hardware) and does not include such items as installation and line lease fees for the terminals. The microcomputer has no additional costs! From a cost effectiveness standpoint, the TRS-80 wins hands-down. A program can purchase three complete microcomputer systems for the cost of two Lear Sigler, ADM-3A video terminals, or FIVE microcomputer systems for the approximate cost of only two DEC Writer IIs.

The arguments are strongly in favor of the microcomputer from virtually all points of view. The microcomputer is:

- More cost effective
- Dedicated to exclusive student use (The G/C/T program doesn't have to be concerned with being pre-empted by some priority task, an experience one cannot escape as a time sharing customer on a large computer).
- A more effective system to teach programming because of its simplicity of operation.
- More easily repaired and maintained.
- A system which can be utilized and taught to children by personnel who are not computer professionals!

The computer has proven itself to be a powerful education device. With the advent of the TRS-80 microcomputer, it is well within the reach of every program for G/C/T children.

The Talcott Mountain Science Center

DANIEL BARSTOW

Every Saturday children from a number of towns throughout Connecticut travel by bus to the top of Talcott Mountain. They have been selected by their elementary and secondary schools as students with special interests or abilities in science. They are participating in a unique and exciting program at the Talcott Mountain Science Center.

Situated at the top of Talcott Mountain ridge in Avon CT, the Talcott Mountain Science Center is a regional center for science education. It has a cluster of science laboratories and classrooms, a staff of specialists, and a wealth of materials for science exploration. Students go to the Science Center to learn by doing—to become involved in real scientific investigation.

In the Saturday program for gifted and talented children, most of the students are working on independent projects. Some students are building electronic gadgets in the radio-electronics lab, or tracking the OSCAR satellite for amateur radio communication from the ham shack. In the evening, astronomy students are using the Science Center's telescopes and developing astrophotographs in the darkroom. For meteorological research the center has a complete weather station, receives weather data by Teletype, and affords a 360° panoramic view of weather conditions throughout central Connecticut. The woods and ponds around the center provide several different ecosystems to explore. In the alternative energy lab, students experiment with solar cells, or build windmills to generate electricity. Of special interest to readers of onComputing, students interested in computers use the Science Center's PDP-11 minicomputer and Apple II microcomputer.

The PDP-11/10 (Digital Equipment Corporation) is a minicomputer capable of serving six independent users at the same time. Users can select from over 100 programs available in the program library, or can write and store their own programs. Though the mainframe of the PDP-11 is located in a corner of the Computer Sciences room, terminals can be connected from several other rooms at the center simply by plugging the terminals into special outlets. Terminals can also be connected to the PDP-11 by telephone (several schools rent computer time from the center).

The Apple II is a single-user microcomputer. It is small and portable enough to be carried from room to room, or even to schools served by the Science Center. The Apple II has a cassette tape recorder and a floppy disk drive (5 inch diskette) for auxiliary memory. The sound and color graphics capabilities are special features of the Apple. The center also has a Heathkit H11 microcomputer, and two digital logic labs. With these facilities, a wide variety of computer-related activities can take place.

Games and Simulations

Many students begin their use of the computers by exploring the various games and simulations available on the computers. With a brief introduction to the procedures for running programs, students are soon independently (and enthusiastically) playing games and simulations such as ASTRO (a simulated lunar landing), POLLUTE (a water pollution experiment), and ANIMAL (in which the computer uses logical thought to guess which animal the user is thinking of). These programs have been selected or developed at the center specifically for their value in science education, and the students clearly enjoy their use.

Many of these programs are used in the context of other Science Center activities. Bill Danielson is Assistant Director of the center, and is in charge of the gifted and talented program. He is also the staff member with primary responsibility for the computer resources. Though his background is in astronomy and meteorology, he has been involved with the center's use of various computers since 1969. Danielson describes the use of the lunar landing program: "Before they play ASTRO, many of the students have seen our videotapes of the Apollo flights, they've observed the moon through our telescopes, they've done a "lost on the moon" activity, and they've created models of lunar craters in the classroom. Then they finish with their own lunar landing. It's a very exciting thing; they really are piloting the lunar module at that point. The visual images are important to help the computer's printout come alive."

Another instructional program is WEATHERWISE. It was written by Danielson to help students learn how to interpret weather data. Several individuals or teams may play. Each team must pilot a ship across the Atlantic Ocean from New York to Iceland. As they trace their programs on maps, students tell the computer in which direction they would like to travel for the next four hours. The computer calculates the new locations, and also reports the local weather conditions. The students must use this information and apply their knowledge of weather maps to avoid the storms over the Atlantic. More than one ship and crew has been lost at sea over the years.

For more advanced meteorology students, there is a program to simulate cloud seeding. Students input control factors such as rain droplet diameter and concentration, and cloud thickness. The computer calculates the amount of rain produced over a given period of time. The students must try to determine the optimum conditions for cloud seeding. John Porter, a staff meteorologist, is revising the program to include the effects of updrafts and downdrafts. He also would like to use Apple high-resolution graphics to show the formation of the clouds and the falling rain.

Computer games and simulations enable students to "experiment" with scientific processes that otherwise might be impossible to experience in real life. Geology students use a program that simulates radioactive decay rates for selected materials (certainly safer than experimentation), and run a program that compresses geological time into one year. Ecology students learn about biological mutation rates with a program that simulates fruit fly (Drosophila) reproduction. This is often done along with experiments

From *onComputing,* Volume 1, Number 3, pages 34–46. Copyright © 1979 by onComputing, Inc. All rights reserved. Reprinted with permission.

with living fruit flies. The computer simulation can compress the months needed for several generations into the brief period of time needed to run the program.

The staff of the Talcott Mountain Science Center has also had to learn how to use these programs. The center employs about fifteen full-time science teachers. Only a few of them had any experience with computers before starting at the center. They are specialists in other fields of science, and have learned about computers on the job. Experience has shown that it is easy for teachers to learn how to use both the PDP-11 and the Apple II computers. It has also been found that the individual staff members acquire different levels of computer literacy, according to their own needs and interests. By now, all staff members at least know how to run programs appropriate to their particular fields of science. Some of the staff also know how to write programs in BASIC, and have worked on programs to serve particular instructional needs. The three staff members who do most of the teaching about computers have had more extensive experience with computers (mostly at the Science Center). In any case, there is a sharing of expertise among the staff members, and the computers are used to some extent in all areas.

Data Analysis

Several of the Science Center's instructional activities take advantage of the computers' abilities to analyze data. In one such activity, students launch helium-filled balloons to determine wind speed and direction. They use a theodolite and stopwatch to accurately measure the changes in balloon positions over time. The students run a special program on the PDP-11 to analyze this data and calculate the observed wind conditions. One group of students used balloon launch data from several locations to study pollution dangers associated with a proposed interstate highway. The study became one of the factors used in a decision to relocate the highway away from a group of water reservoirs.

The center has solar prominence and sunspot telescopes, as well as a solar spectrometer. Various programs are used to analyze sunspot activity and calculate solar temperatures. Programs like these help students interpret and understand research data.

Chronobiology is the study of biological processes as they rhythmically change with time. For example, temperature in human bodies varies according to regular patterns throughout the day; plants blossom in annual rhythms; the human menstrual cycle is approximately twenty-eight days. (Chronobiology should not be confused with the pseudoscience of "biorhythms.") Science Center students use a range of equipment to study rhythms in plants, animals, and humans. The computer is an essential tool for this study.

Dr. Donald P. LaSalle, Director of the Talcott Mountain Science Center, has been a pioneer in educating children to use biological measurements to learn about their own circadian (daily) rhythms. LaSalle comments, "The computer is to chronobiology as the telescope was to astronomy. Galileo's telescope enabled him to see detail precise enough to recognize the true nature of the solar system. The computer enables us to study large enough amounts of biological data to recognize subtle changes. Before computers, sophisticated chronobiological research was virtually impossible. With computers, even elementary school children can take their own biological measurements and run the data analysis programs." This information can help students recognize their own daily rhythms, and understand how these rhythms affect their daily lives.

Student Programs

After a few sessions working with the computers, most students are eager to learn how to write their own programs. In the Saturday program, students may choose to participate in a six to twelve week session on computer programming. The emphasis is on applying the computers to the students' work at the center.

BASIC is the programming language used for both the PDP-11 and the Apple II (some of the more advanced students opt to learn machine language as well). Though BASIC has some limitations, students find it easy to learn and apply.

The staff has found it helpful to use a "read-modify-write" sequence to teach particular commands. In the case of FOR-NEXT loops [a loop is a type of computer language structure in which the computer repeatedly cycles through a series of instructions], for example, students first read a simple program which contains a FOR-NEXT loop, while the teacher demonstrates its use. Next, they observe the effects of modifications in the program, such as moving the NEXT statement. Finally, the students write their own programs using FOR-NEXT loops. This sequence can be applied to any command, and helps students progress from the concrete to the abstract.

Some early programs include: mathematical calculations such as area and average, graphic manipulations, word quizzes, a coin toss simulation, and conversions such as Fahrenheit to Celsius. It takes just a few weeks of practice for students to feel comfortable writing programs in BASIC.

As students begin to work on longer programs, the staff has found it important to teach structured programming techniques. (Structured programming refers to a particular approach, and techniques, for writing programs that are logically organized and clear to understand—human readability is as important as computer readability. Students are shown how to use a "top-down design," which involves breaking the overall program design into smaller, more manageable sections. These sections can serve as the basis for "block-structured" programs. REMarks are used to indicate the flow of control in the programs, and explain the operations of sections that might be confusing or unclear. A heading is required on all student programs to indicate the name of the program, the author, and a "save until" date (student programs are periodically purged). Preferably the heading also includes a list of variables and a brief description of what the program does. There are some limitations on the use of "true" structured programming techniques, because BASIC lacks certain control structures such as WHILE and UNTIL loops. Nevertheless, the staff has found it essential to specifically teach a logical approach to program design. Otherwise student programming projects become confusing and unmanageable.

Many of the students' programs are related to other areas of interest. A student who has had some experience flying airplanes is presently writing a game program which involves a simulation of the airplane's controls. Various different scenarios can take place, and a successful flight requires careful calculation of flight direction, wind speed, fuel consumption, and altitude.

An ecology student, who had been studying about the predator-prey relationship of rabbits and foxes, recently made a computerized model of a forest. The program calculated the total populations of rabbits and foxes, based on such factors as initial populations, food available in the forest, rainfall, and natural disasters. Defining these interrelationships in mathematical terms became the focal point for a careful ecological analysis.

Lisa Barnhart, a high school student, used the computer to study the correlation between crime and weather. She based her study on crime statistics from the Hartford police, the state police, and the FBI, along with weather data for a five year period. Quoting from her summary report: "The temperature had the greatest influence on crime. It had its largest effect on larceny, then in descending order, on aggravated assault, burglary, and robbery . . . Precipitation, strangely enough, had a great effect on two crimes: robbery and burglary. I had anticipated a lower crime rate when there was more precipitation."

Michael Dowling, another Science Center student, is working on a program which will calculate the orbits of planets, comets, asteroids, and other orbiting objects, using the Laplacian method. The calculations are based on position data from three precise observations through the center's 32-centimeter reflector telescope.

A geology student wrote a program to identify rocks based on descriptive data. By asking questions about rock characteristics, the program is able to identify several types of rocks. The classification scheme underwent several revisions as the student learned more about rock classification systems.

The color graphics of the Apple II offer powerful capabilities that the students enjoy exploring. In some cases the students experiment with kinetic art, manipulating the graphic images under program control. One group of students used the Apple to display a graphic image of the sun and indicate the placement of sunspots according to daily observations. Their program attempted to duplicate the correct motion of the sunspots.

A project not yet completed is the storage of star position data in the Apple II. Users will be able to specify particular areas of the sky, and the Apple II will use high-resolution graphics to plot the appropriate stars in their correct positions. When it is completed, the Apple II can be easily carried to the center's observatory to facilitate stargazing.

Though some of these programs may seem fairly sophisticated, few of the students had any experience with computers before going to the Talcott Mountain Science Center. Once students are familiar with a programming language such as BASIC, writing programs is not a difficult process. The focus of their work usually is on designing the algorithms, or procedures, which form the basis for the program. Thus the emphasis is primarily on scientific understanding of the problem. Actually writing the program (and implementing, testing, and debugging) involves time and work, but is rarely too difficult for the student to handle.

The major problem with computer use at the Science Center is providing sufficient computer access time to the students. In spite of the availability of six PDP-11 terminals, the Apple II, and a few other computer resources, demand for computer time is higher than the supply. There are several aspects to the solution, none of which completely alleviate the problem. The computer programming class has a lower student capacity than some of the other classes—twenty students (with two teachers) is a maximum. With the Apple, and four or five terminals to the PDP-11 (some terminals are usually used by other classes), this provides a student/computer ratio of about four to one. Sometimes students work in groups, and there is a sign-up procedure to take turns. The center also uses student tutors to help answer questions, both between turns and at the terminals. It's important to remember that the off-line time is also vital to the program development process. Students need to study the problem and design the program before sitting at the terminal. Also, corrections and improvements can often be seen more clearly when the programmer gets away from the immediacy of the keyboard and printout.

Hardware Projects

"We definitely are software oriented," reports Danielson. "It's clearly easier for students to write their own programs than to design hardware construction projects. However, a number of our students have gotten involved with hardware design and use our electronics lab to build computing machines."

The Science Center has a few commercial digital logic labs, which help introduce students to the theory and practice of digital design. The center's first microcomputer was a Heathkit H11, a kit built by a group of students. Other students have built computer kits for personal use. With the help of Bob Judd and Al Vitiello, the center's electronics specialists, some students have designed and built their own computers.

Tom Birdsall is a high school student from Andover CT. He describes his experiences with "homebrew" computers: "Two and a half years ago I felt that I wanted to learn how computers worked by building a microcomputer—not from a kit, but from scratch. My computer is based on an 8080 microprocessor, and has 1 kilobyte of programmable memory. The first version was built on a protoboard. My second version features a case and a switch register for input/output. Though it is currently limited to mathematical manipulations, in the future I hope to use my computer to synthesize music, control remote equipment, and, above all, as a general-purpose home computer."

The Saturday program is certainly the most intensive experience students have with computers at the Science Center. Yet it should not be thought that only the "gifted and talented" can learn how to use computers. Throughout the week the center offers numerous other computer-related services to educational institutions at all levels. (It is important to keep in mind that the computer is just one of many instructional and investigative tools used at the Science Center.)

Computer Literacy

Some schools contract the center to teach basic computer literacy, either in the schools or at the center. The purpose is to expose students to the capabilities of computers, and to provide some experience in their use. The minimum is a three session sequence, which includes an introduction to computers, a chance to play some computer games and simulations, and some introductory experiences with BASIC. Several schools rent computer time from the center, with the benefit of access to its wealth of science-related programs. The Board of Education in Southington CT recently purchased an Apple computer, and has contracted the services of the Science Center to help them learn how to use it in their schools. Staff member John Porter travels twice a week to Southington to teach students and teachers how to use the Apple and how to program in BASIC. He also offers suggestions on how to apply the computer, especially in science education. Porter enjoys using the computers, and his enthusiasm is an important factor in turning kinds on to computers.

Research

For several years the center has been involved with research on solar energy. Two of the buildings at the center have solar panels providing some of their energy needs. The new "Solar-heated Chronobiology and Appropriate Technology Laboratory" has several banks of active solar collectors on its roof. They provide all of the hot water and 50% of the space heating energy resources. In addition to collecting information about the efficiency of these panels, the center also operates solar instruments that measure direct and diffuse sunlight throughout each day. The data from these measurements is being collected and analyzed by computer to help determine the potential for solar energy in this area.

Other agencies have contracted the use of the Science Center's computer for analysis of research data. The American Radio Relay League conducted a study of its members (amateur radio operators) to determine their interests, activities, and purchases related to ham radio. Over 3000 responses to the questionnaire were received, and analyzed at the center by the PDP-11. The PDP-11 is used for most of the research because of the large capacity of its two hard disk drives (2.4 megabytes of information per disk).

Administrative Applications

There also are administrative uses for the computers. The mailing list for the center's news bulletins has been computerized. This certainly speeds up preparation of address labels. It also enables selected notices to be sent to specific people, such as elementary school principals or science department chairpersons. The computer is also being used to keep track of the center's inventory of materials. Danielson is working on a staff scheduling module, which should alleviate some of the problems involved in the complicated process of matching school needs, staff abilities, and Science Center resources. These administrative applications are run for the most part at times when students are not using the computers.

The Future

What about the future? The Talcott Mountain Science Center is always exploring new areas of science education. Because personal computers are becoming more popular, the center is likely to offer evening classes on the use of microcomputers and how to program in BASIC. There may also be classes in digital logic for those interested in hardware experimentation.

Since many public schools are also buying microcomputers, the center will be providing special training courses for teachers to learn how to use computers. This will include demonstrations of the computer's applications in science education. Most of the science related programs developed at the center can be used in other schools, even if they do not have the extensive resources available at the center.

Some hardware acquisitions are likely. A light pen, bit pad, or other graphic input device would facilitate the use of Apple graphics. The Apple's existing 16 kilobyte capacity of programmable memory is inadequate for some applications, so another memory card will probably be purchased to expand capacity to 32 kilobytes. Since a number of the center's instruments are analog devices, analog-to-digital (A/D) converters are needed to interface these instruments with either the PDP-11 or the Apple II. (Analog refers to continuous change, such as a moving second hand on a clock. Digital refers to counting by specific numerical increments, such as the changing time in digital watch displays. Most computers can process data only in digital form.) The center may also explore the use of speech generation and speech recognition devices.

The most important aspect of improving the educational uses of the computers is new software. Some programs will be purchased, especially since the boom in microcomputer use is motivating software companies to make available more programs. However, it has been the Science Center's experience that programs applicable to its needs are best developed by the staff.

Bob Judd, head of the ham radio station, is writing a program to use the Apple II's speaker for a Morse code drill. Another staff member is writing a program to display various electronic circuits and a chart of calculated values at different points in the circuits. This is in response to some student confusion related to voltage and current in circuits. Staff members are also working on a program to plot data from weather stations on a computer-generated map of the United States (based on map generation programs in *BYTE* magazine, May and June 1979). The computer has even been programmed to perform magic, as part of an evening course on "Science and Magic" [See "Magic for Your Micro" in the Fall 1979 onComputing ... ed]

The computer is definitely having an impact on science education. At the Talcott Mountain Science Center, computers have been used for ten years, and have become an integral part of the way science is taught. The computer is a powerful, multipurpose tool that expands the intellectual reach of both students and teachers. To quote Dr. LaSalle, "The basic philosophy of the Science Center is to involve students in scientific exploration. We have found the computer to be an essential tool in this exploration."

Dan Barstow was a teacher at the Talcott Mountain Science Center. A fluent speaker of Spanish, he currently is Project Director for a bilingual gifted and talented program in Hartford. He also masquerades as Merlyn the magician, performing feats of prestidigitation.

Computeronics: A Course in Computer Literacy

PRISTEN BIRD

Computer education is now considered a priority in education for all students, especially for the gifted and talented. The National Council of Teachers of Mathematics surveyed educators, parents, and community members to determine priorities for school mathematics in the 1980's. The majority said that the curriculum should emphasize problem solving, while 90% said that the types of problems taught should be those that can be solved by computer. Development of computer literacy was outranked only by mathematical applications. This emphasis is particularly pertinent for the 2.5 million gifted and talented students in the United States. In addition, the increasing availability of computers—particularly of microcomputers—in homes and schools is encouraging educators to find ways of maximizing their use in the classroom.

In 1974, the Gifted Child Project in Tallahassee, Florida, was federally funded by an ESEA Title IV-C grant to create innovative instruction. Between mid-1974 and 1981, the Gifted Child Project developed two courses for gifted and high achieving sixth and seventh grade students—Thinkology and Computeronics. This article will focus on the Computeronics program.

Computeronics was developed to provide students with a perspective about computers—to learn a simple programming language, to use computers to solve problems, and to have an accurate conception of the role of computers within their lives and in society in general. Fully evaluated, federally supported, and approved by the US Office of Education, Computeronics is a 35 to 40 hour course in computer programming, problem solving, and literacy. To meet the goals of the project, the course was divided into two modules: Problem Solving with Computers and Computers in Society.

Problem Solving with Computers

This module consists of approximately 24 hours of class time. Students learn a computer langauge, how to write programs, and how to use the computer as a problem solving tool. BASIC was chosen as the language because it is easy to learn and is also widely used. Therefore, the course is easily adaptable to many different computer systems.

Students use the computer to apply their programming skills in solving word problems. They spend as much as 80% of their time reading, completing activities, and writing computer programs at their desks or at typewriters. The remaining time is spent in hands-on activities at the computer.

Sidetrips, a book of over 40 brain teasers and problems, provides additional activities for students who finish early. The activities require that students apply their programming skills to problem situations. Students first determine if a computer is needed to solve the problem, then they go on to devise solutions. A given program may need debugging or may require skills in graphics which students have not yet learned. Or, students may be asked to study and decode a program to determine its purpose.

Computers in Society

The goals of this module focus on information rather than on skills. Students learn about the history of computers, their present and future uses, and computer-related careers. *Computer View,* the student text, employs a magazine format that includes articles, photographs, advertisements, puzzles, a glossary, and a feature section on computer history. This module consists of approximately 15 hours of class time.

Computeronics was developed by teachers, instructional designers, computer experts, and students. From the beginning, students pilot tested the materials. They practiced skills, took mastery tests, and checked their work, and, if successful, continued. If not, they tried a second version of the mastery test. The mastery approach remains an essential part of the course.

Teacher Roles and Training

The management system built into student lesson books, activities, and mastery answer book allows students to move at their own pace. Teachers can be directive, facilitative, or both. One teacher commented, "My summer school class contains fifth through twelfth graders. They're all working at different levels, but they can move through the course. Some need almost no help; others require a lot of my support."

To adopt the course, teachers need the teacher materials and two days of inservice training. They are trained to use the course as teachers and as students. They work through the lessons and activities, experiencing the joys and frustrations of a successful RUN or a nagging "Syntax Error." After completing the inservice, one teacher remarked, "I figured if my sixth graders can learn this, I can too. I was exhausted afterwards, but I walked about three inches off the ground."

Conclusion

As of mid-1981, Computeronics is the only computer project that is officially approved by the US Office of Education to join the National Diffusion Network. The purpose of the Network is to support the distribution of projects that have proven their effectiveness. Computeronics will continue to provide three important services: (a) inservice training for teachers, (b) cost-effective materials, and (c) assistance to schools to meet local needs and state/federal mandates.

For more information about Computeronics contact: Pristen Bird, Director, Gifted Child Project, 2757 W. Pensacola Street, Tallahassee, Florida 32304.

5 Getting Started

A school, a district, or a state may be convinced that computers are needed in the system but no one is really sure how to select appropriate equipment, determine an adequate budget, or evaluate the needs of the children and teaching staff. The articles in this section describe some possible approaches.

In *Bringing Microcomputers Into Schools,* George Ropes and Henry Gaylord discuss how to get funding, where to look for software, what equipment to buy, and how to start a microcomputer curriculum.

Statewide Educational Computer Systems: The Many Considerations by Kevin Hausmann describes the Minnesota Educational Computing Consortium's plan for statewide support and acquisition for educational microcomputers. A list of microcomputer systems and minimum specifications used to evaluate them is included.

There are many questions a school system must ask when it first beings to consider the purchase of a small computer. In *Van Helps Schools Select the Right Computer,* Betsy Staples provides a systematic approach to evaluating needs. The Pennsylvania Department of Education has equipped a van with an array of audiovisual paraphernalia and a media specialist. The van travels around the state helping districts evaluate their needs. A sample checklist and a series of 17 questions to ask about a computer are included.

Bringing Microcomputers Into Schools

How to get funding and where to look for software, what equipment to buy, how to start a microcomputer curriculum? These are some of the questions addressed in this article.

George Ropes
Henry Gaylord
Box 53J
Goldens Bridge NY 10526

Schools are on the threshold of the microcomputer age. Not only high schools, but also junior highs and elementary schools can share in the many benefits microcomputers bring to education. Educators are asking what they can do, what they should do, to make microcomputing capacity available to their students. This article provides the information and procedures educators need to leave "ground zero" and join the ranks of those who have already found how much a few microcomputers, or even a single micro, mean to a school.

Objectives

Those who have worked with microcomputers are strongly of the opinion that no other single piece of equipment can do as much for education. The nearly unanimous conclusion is that micros motivate students to a remarkable extent.

They are unexcelled at teaching logical thinking. They provide frequent opportunities for students to demonstrate their creativity. They are valuable in all curricular areas, not just in mathematics. They provide effective, individualized review and practice, with immediate feedback on performance. Learning about microcomputers and how to use them may be the most valuable part of a student's schooling.

A school contemplating a microcomputer program should set its sights on accomplishing at least one of the following goals:

Increasing students' computer literacy
Teaching students to write programs
Providing opportunities for learning through simulations and games
Teaching subject matter, such as mathematics, science, social studies
Building skills in vocabulary, spelling, computation, shape recognition
Drill and practice on basics

The broader the scope and objectives you have in mind, the more thorough the preparations and training should be. Once you have an idea of the direction to take, two considerations have priority: sources of funds and choice of microcomputer.

Obtaining Funding

Schools have found a number of different answers to the problem of funding. Some boards of education have put money for microcomputers into their budgets. Some school districts have applied for and obtained federal grants under Title IV-C sections authorizing funds for career training or for gifted and talented students. Others have convinced the local Parent-Teacher Association to appropriate money for one or more microcomputers. When the cost-effectiveness of a microcomputer is compared to that of a color television, videotape equipment or even to a district's service vehicles, funding problems are seen in better perspective.

Choice of Equipment

Deciding on which microcomputer or microcomputers to purchase is important, but not necessarily critical. There are differences between micros as there are between automobiles, but with each of the popular makes you can expect years of dependable service.

You shouldn't put off purchasing a micro in the hope that a sharp price decrease is just around the corner. Few people believe that microcomputer prices will come down the way calculator prices fell over the last few years. You can expect the next round of microcomputer models to have greater capabilities at or near current prices. If you want your students to start getting to know and use micros, the time to begin is now!

For most schools, we feel that decisions on equipment should favor the solution that accommodates the largest number of students. The experience of sitting down at the keyboard and operating the microcomputer is crucial to learning.

We recommend that you have at least 8K (approximately 8000 characters) of addressable memory. You will do well to have a relatively full version of BASIC, the language used by virtually every microcomputer. Your micro should be able to handle decimal numbers, not just integers. Choose color capability if programs involving designs and patterns are important to you.

Current (April 1980) prices for the three best-selling models are approximately as follows:

$850—Radio Shack TRS-80 Level II 16K. (The TRS-80 Level I 4K costs $499. If this is the only one you can afford, plan to spend $250 to $350 to upgrade it to Level II.)
$530—Commodore PET 8K. (Based on cur-

Henry Gaylord is assistant head, Cisqua School, Bedford NY. He has taught computer programming throughout the school, for Mercy College and for the Chappaqua Adult School.

George Ropes is consultant on math for the gifted, Putnam/Northern Westchester B.O.C.E.S., adjunct professor at Fairleigh Dickinson University, Teaneck NJ, and elementary math coordinator, Edgemont School District, Scarsdale NY.

From *Microcomputing*, June 1980, pages 104–106. Copyright © 1980 by Kilobaud Microcomputing. All rights reserved. Reprinted with permission.

rent 3-for-2 offer to schools. No tape recorder included.)

$1195—Apple II with Integer BASIC or Applesoft BASIC 16K. (No tape recorder or monitor included.)

Check out the micros available at nearby outlets. Check on the experience of other schools in your vicinity. A knowledgeable and interested person at a local store can be a valuable aid to your microcomputing program.

Learning the Language

After considering funding and the microcomputer model, you should examine how to use the microcomputer(s) in your school. At least one person should be familiar with the BASIC language, which consists of about 30 words, a dozen symbols, two dozen functions, punctuation marks and a few rules for writing statements.

Mastery of BASIC takes lots of time and practice. However, the ability to write a simple program in BASIC or to understand uncomplicated programs written by others can be acquired relatively quickly. Knowledge of BASIC is valuable for introducing the language to students and making minor modifications in commercially available programs. It is often useful to change a program so it is easier or more difficult or more appropriate for a given student or group of students.

If someone in your school already knows BASIC, you're well on your way to microcomputer capability. However, there are minor differences between BASIC as implemented on time-sharing terminals and the BASIC of microcomputers, as well as between the versions of BASIC on various makes of microcomputers.

Start-up and editing procedures differ slightly also. Some functions are called by different names: the ENTER key on one machine might be RETURN on another. Generally, a little practice and a few references to the manual are all it takes to smooth the transition.

To learn BASIC from scratch is not a mountainous undertaking. First, you must obtain a good manual. The best manual on the market is *BASIC Computer Language* by David A. Lien. It is published by Radio Shack and applies specifically to the version of BASIC used on the TRS-80 Level I. This version of BASIC does not include some of the commands and statements found in the Level II version of BASIC. However, in a separate volume, *Learning Level II* (Compusoft Publishing), the author provides all the changes—line by line and page by page—to convert the earlier manual into one that teaches you Level II BASIC on the TRS-80.

If you are using a PET microcomputer, get *Hands on BASIC with a PET* by Herbert D. Peckham, McGraw-Hill. For those using BASIC on an Apple II computer, the manufacturer's *Apple BASIC Programming Manual* is the answer.

But whatever microcomputer you have and whatever manual you use, best results come from sitting down at the computer and working through the material page by page. Just as in most other learning experiences, you learn fastest and best by doing. This is not to denigrate courses that teach computer programming.

We have taught hundreds of students of all ages and have found that the greatest amount of learning takes place when a student practices what he learns on the computer in the days between class meetings. Fortunately for teachers, microcomputers are portable. They can be easily transported from classroom to classroom or taken home.

Does every teacher at a school need to know BASIC? Certainly not. The more who do learn BASIC, the better, but microcomputer use can be extensive in classrooms where teachers know nothing of the language.

Creative Computing, Box 789-M, Morristown NJ 07960.
Hayden Book Co., 50 Essex St., Rochelle Park NJ 07662.
Instant Software, Inc., Peterborough NH 03458
Micro Learningware, Box 2134, Mankato MN 56001.
Minnesota Educational Computing Consortium, 2520 Broadway Dr., Lauderdale MN 55113.
National Coordinating Center for Curriculum Development, State University at Stony Brook, Stony Brook NY 11794.
Program Design, Inc., 11 Idar Court, Greenwich CT 06830.
Programma International, Inc., 3400 Wilshire Blvd., Los Angeles CA 90010.
The Program Store, 4200 Wisconsin Ave., N.W., Washington DC 20016.
Queue, 5 Chapel Hill Dr., Fairfield CT 06432.
Radio Shack (contact nearest store).
Software Exchange, 6 South Street, Milford NH 03055.

Table 1. Sources of cassette tapes in the educational field.

Every teacher must know how to operate the microcomputer and how to load programs into it from cassette tapes. Incidentally, the teacher need never fear that microcomputers will eventually "take over" the classroom. Actually, the more computers, the more important the teacher becomes.

Programs on Cassettes

You can connect each microcomputer to a standard cassette player, which is used to provide long-term storage of programs. The process of entering programs via the cassette player is comparable in difficulty to running a film projector. After a few minutes' instruction and an hour's practice, the task becomes routine. It involves positioning the tape to the start of a particular program, typing a "load" command on the keyboard, playing the cassette and waiting until the computer indicates that loading is finished.

Programs on cassette tapes are referred to as "software." The amount of software available for use in schools is growing; new tapes are coming on the market every month. Unfortunately, program quality varies; some tapes contain programs that are excellent; some are fair; some poor. It is not easy to get responsible opinions about software in advance of purchase.

Among the types of educational programs available are:

Demonstrations of math processes
IQ building with synonyms, antonyms
Work with foreign languages
Spelling, vocabulary
Games requiring logical thinking
Simulations
Recognition of shapes, letters
Arithmetic drill and practice
Graphic designs

From time to time, this magazine and others in the field publish programs that can be used or adapted for use in schools. It pays to keep abreast of developments by subscribing to two or three of the leading periodicals. The computer departments of a few universities have started to make available information about educational programs at all levels, as have education departments in a few states.

Table 1 lists companies and organizations that distribute educational software. The listing is representative only; it does not purport to include every company that offers programs you might use.

Be sure any programs you order are for your particular microcomputer. Programs are not interchangeable between makes of computers. A TRS-80 Level I program cannot be used on Level II. Similarly, programs for Apple II with Integer BASIC don't work on Apple II with Applesoft BASIC.

The demand for educationally based microcomputer programs is such that new programs originated at your school may be useful to many other school systems. Someone on your staff—or among your students—may develop better programs than some of those that exist today.

Utilization

Given a school with one or more microcomputers, given one or more teachers who know BASIC and given a set of good programs, what is the best way to deploy these resources? How is capability transferred into day-to-day operation? There are several possible procedures.

Microcomputers may be placed in a given location where classes can be scheduled for specific computer programming lessons. Individuals can come to that location for specified periods of time, usually on a "sign up in advance" basis.

Must someone be present at all times to supervise students? Some schools have made arrangements for such supervision, but many have found it unnecessary. Microcomputers are relatively foolproof. Provided no more than two students are assigned to one micro at one time, experience has shown that they work well by themselves.

Another common procedure is to use an audiovisual cart to move a microcomputer from room to room on a scheduled basis. This allows the teacher to use the computer to demonstrate a given lesson or to assign students 10 or 15 minutes each on the computer. Different students can work with different programs. Students, even as young as second-graders, quickly learn how to insert and run tapes.

A network controller for the TRS-80 directs the loading of programs into as many as 16 microcomputers in a single classroom. Such a setup permits working with a class as a unit and also lets each class member interact individually.

Junior high and elementary schools have found it advantageous to have a knowledgeable high school student work with their students. The high school student can teach BASIC or act as a resource person for a "computer club." Parents with some background in computing can be of assistance at all levels. Information on special programs or counsel on long-range objectives may often be obtained from a consultant in the field.

The Future

Each school needs to work out the procedure or combination of procedures for microcomputer implementation that best suits its particular needs. Whatever the needs are today, it is safe to predict that they will be different and greater tomorrow. The numbers of students who are turned on by microcomputing will convince you to get more micros and expand your program until the majority of students acquire the computer capability that will virtually be a sine qua non for life in the 21st century. ■

Statewide Educational Computer Systems The Many Considerations

Kevin Hausmann

This article describes the Minnesota Educational Computing Consortium's (MECC) plan for statewide support and acquisition for educational microcomputers and focuses on the need, development and implementation of a plan.

Although the utilization of microcomputers in education is relatively new, within the next few years we can expect an exponential increase in the number of microcomputers sold to educational institutions. The growth is being spawned by a number of factors including the decreasing cost of microcomputers coupled with their increasing capabilities and the rapid growth of their use in a variety of fields throughout the country. Another important factor is the microcomputers' independence from a mainframe system which increases its portability and eliminates many communications-related problems, as well as the elimination of the "limiting rules" needed on central systems.

The potential of microcomputer applications' has attracted the attention of hundreds of vendors ranging from garage hobbyists to major mainframe companies as evidenced by the attendee's at personal computing fairs and National Computing Conferences. The resulting number of different systems makes it very difficult to stay abreast of current developments. Microcomputers will also follow the paths of other new technologies, meaning, many of the current microcomputer manufacturers may go out of business within a year or so, making it very important for microcomputer purchasers to be aware of manufacturer and vendor stability.

Microcomputers will also follow the paths of other new technologies, meaning, many of the current microcomputer manufacturers may go out of business within a year or so.

Many educators view the microcomputer as a panacea for a variety of educational ills resulting in expectations which are greater than system capabilities. In many cases, the use of systems will be impeded by a lack of hardware or software features. In addition, applications software development and instructional support will not keep pace with the initial movement to microcomputer usage.

In order to meet the needs and address the problems defined above, MECC set up a special task force to accomplish the following:

1. To conduct a survey for assessing the current and future microcomputer uses and needs of MECC users.
2. To determine the strengths and weaknesses of microcomputer utilization in various instructional computing modes and environments.
3. To provide demonstrations of microcomputer use for instructional purposes.
4. To coordinate and disseminate information regarding pilot programs using microcomputers.
5. To prepare recommendations regarding the potential for large scale acquisition and

Kevin Hausmann, Instructional Coordinator, Minnesota Educational Computing Consortium, 2520 Broadway Dr., St. Paul, MN 55113.

From *Creative Computing,* Volume 5, Number 9, pages 82–85. Copyright © 1979 by Creative Computing. All rights reserved.
Reprinted with permission.

Statewide, con't...

utilization of microcomputers and the appropriate roles and responsibilities for MECC.

To begin addressing these objectives, the task force divided the tasks into three components: hardware, systems software, and applications software. Several Minnesota vendors were contacted and asked to supply systems for examination, evaluation and experimentation. Fourteen systems were evaluated by the task force.

Microcomputer Systems Evaluated

ALTAIR ATTACHE
ALTAIR 88-1301
ALTAIR MULTI-USER
ASTRAL 2000
IMSAI VDF - 80/1000
NCR 7200
OLIVETTI P6060
PET 2001
POLYMORPAICS 8813
PROCESSOR-TECH SOL
RADIO SHACK TRS-80
APPLE II (Integer BASIC)
TEKTRONIX 4051
TERAK RT-11

As a starting point, the task force defined a "minimal educational system." The minimal educational system must have:

a) A microprocessor.
b) I/O device(s). The system must include an ASCII keyboard and printer and/or monitor.
c) A permanent file storage device. This can be of the form: floppy disk, hard disk, etc.
d) A vendor supplied operating system.
e) The BASIC programming language must be supported.
f) At least 12K of user memory must be available. This excludes memory space required for the operating system and the language processor(s).
g) All components, software and hardware must be documented. This must include instructions on the operating system, a language manual and setup and maintenance instructions of the system.

Only those systems which met these specifications were considered. This definition also helped vendors identify which features are required in a system for educational purposes.

In looking at software features, the task force considered both whether or not a system had a particular feature as well as how important that feature was to users. Over fifty software features were identified and classified as to importance for two types of usage: Computer Science/programming and applications/programming.

The computer science/programming classification was defined as the use of a microcomputer system to meet the needs of secondary and post-secondary computer science instruction involving such topics as advanced programming, operating systems, compilers and assembly languages. This is a system used primarily by those who are interested in studying the computer system itself, therefore software flexibility is important.

The applications / programming classification includes using the microcomputer for running application programs as well as writing and running simple BASIC programs for problem solving in elementary and secondary schools. This mode requires the capability to run programs which generally range from 8 to 32K in core requirements. Since this system would also be used to run programs similar to the library programs on the existing timeshare system, downloading capability is highly desirable, if not essential.

The applications/programming classification includes using the microcomputer for running application programs as well as writing and running simple BASIC programs for problem solving in elementary and secondary schools.

In trying to evaluate microcomputers against these classifications of use, each feature was given a rating of essential, desirable, or not necessary for the two classifications as defined above. Each microcomputer was given a yes or no score on each software feature. By combining the importance scores with the yes-no scores, it was possible to give each microcomputer a software feature score for both classifications of use.

Since BASIC is the most often used language, the task force also attempted to evaluate the microcomputer's BASIC language features and capabilities which they deemed important. Test scripts were prepared and run on each of the fourteen systems.

Ten scripts were prepared. The scripts were divided into two catagories, those that tested BASIC language features, and those which tested performance. BASIC features scripts included sequential file handling, random access file handling, chaining, time function, string functions, matrix operations, and formatted output. The BASIC performance scripts included time required to complete computation (calculate number of primes from 1 to 2000), number of mathematical functions available; and time required to generate and sort 100 numbers.

In looking at hardware, some 40 features were defined and each system was rated against these. Some of the typical features included were: K-bytes of RAM, ROM, or PROM; available user memory; chip type; add time; availability of RS-232 interface and a real-time clock.

In addition to working with vendors, the task force evaluation of systems included a user survey which was developed and administered to teachers in the state who were using microcomputers in their classes. Questions dealt with types of usage, features of the microcomputers which were particularly desirable, and problems which were encountered with their systems. The major weaknesses of microcomputer systems currently in use seems to be the availability of CAI languages, ability to perform repetitive calculations, and the storage and movement of large data files. However, the majority of instructional computing can be accomplished quite well using microcomputers.

The major weaknesses of microcomputer systems currently in use seems to be the availability of CAI languages, ability to perform repetitive calculations, and the storage and movement of large data files.

Once done with defining needs, collecting data on microcomputer systems, and surveying current microcomputer users the task force made the following recommendations regarding microcomputers:

#1 State Contract:
One specific microcomputer system should be available to all Minnesota educationally-related agencies through a state contract.

#2 Support:
Instructional service support

Statewide, con't...

for selected microcomputers should be defined and increased to the same level as is currently available for large timeshare systems.

#3 Microcomputer Technology: MECC should continue to analyze and evaluate microcomputer hardware and software technology, and disseminate information to the Minnesota educational community.

On October 15, 1978, MECC and APPLE Computer, Inc. signed a contract for APPLE's 32K, disc-based Applesoft microcomputer system.

Educational computing service agencies must develop plans early if they are to cope with the fast-growing microcomputer industry.

MECC anticipates about 400-500 units to be sold by APPLE through MECC to educational users in Minnesota during the current year.

Educational computing service agencies must develop plans early if they are to cope with the fast-growing microcomputer industry. There is absolutely no indication of this growth trend slowing down. The fear that service agencies will no longer be needed is totally unfounded. Users will still need the software support that they have in the past. However, service agencies will have to redefine what they call "service" or "support." There are four major areas of microcomputer support that must be considered.

1. Purchase, installation, maintenance, and documentation of the system.
2. Training in system operations, use of application packages and programming languages.
3. Acquisition, conversion, development, maintenance, documentation and dissemination of applications packages.
4. Response to questions, problems, and requests regarding microcomputers.

It is hoped that the comments related here will be of help to agencies faced with the problem of servicing microcomputers.

Note: A task force report was printed by MECC, and contains the Task Force research, evaluation, recommendation and the invitation for Bid for microcomputer systems. A second revision of this report will be available soon. Contact MECC, 2520 Broadway Drive, Lauderdale, Minnesota 55113 for more information. ☐

MECC

The Minnesota Educational Computing Consortium (MECC) was created in 1972 out of concern by the governor and legislature that educational computing needed a central source of coordination for planning, and a mechanism to insure that all educational institutions in the state would have equal opportunity of access to computing services for both instructional and administrative programs. The Consortium's membership includes the University of Minnesota (5 campuses), The Minnesota State University System (7 campuses), the Minnesota Community College System (18 campuses), the Minnesota Department of Education (representing the state's 436 independent school districts) and the Minnesota Department of Administration. Minnesota is the only state in the country having a central organization for coordinating educational computing activities across all levels of education.

The MECC Instructional Services Division offers a variety of services to consortium members. A technical staff operates the largest of Minnesota's computers dedicated to instructional computing, a Control Data CYBER 73 time-sharing system. The MECC Timeshare System is currently configured for 375 user ports and serves about 1100 interactive terminals located in schools and colleges across the state. A large multiplexing communications network provides the route by which MECC users access the Timeshare System, whether they are a few miles from the Minneapolis-St. Paul computer center or hundreds of miles away near the Canadian border. The MECC User Services staff of instructional coordinators helps users learn to make better use of the computer by visiting school and college sites, conducting workshops, providing over-the-phone consulting service, publishing newsletters and producing written documentation for programs in the MECC Timeshare System's central library.

Van Helps Schools Select The Right Computer

Betsy Staples

Which is the best computer? Which software should we buy? What can we actually do with a computer? How can we teach people to use it? Who will fix it when it breaks?

These are only a few of the questions that confront a school system when it first begins to consider the purchase of a small computer or computers for classroom use. Sometimes the questions are never answered adequately. Sometimes teachers and students are unhappy with computer hardware and software chosen in a haphazard fashion.

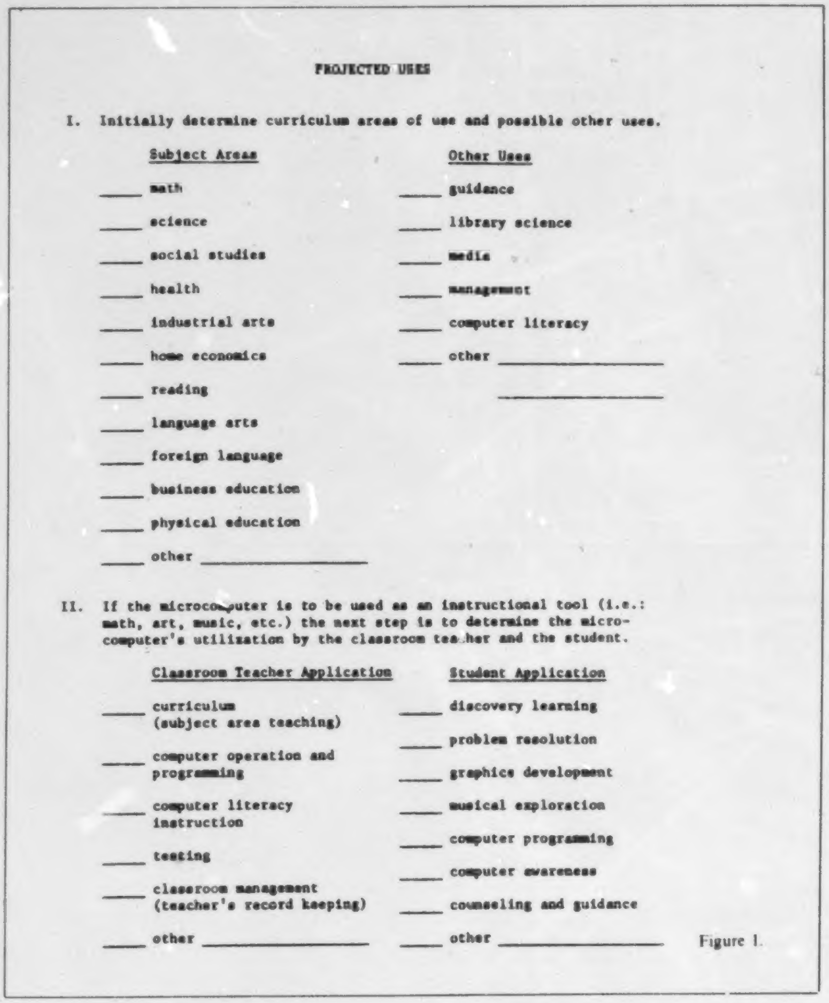
What seems to be needed is a systematic approach to the selection of hardware and software for classroom use. The Pennsylvania Department of Education offers just such an approach to almost 500 schools in the Commonwealth in the form of the Multi Media Training Van, a vehicle which provides information and training in everything from writing to photography to computers. The van, staffed by Media Specialist Shirley Douglas, travels around the state visiting various school districts which have requested its services. The project is currently a joint effort of the Pennsylvania Department of Education and a Title IV-C project at the Colonial Northampton Intermediate Unit, Nazareth, PA. The program is coordinated through the state's Intermediate Unit Instructional Materials Service department.

In addition to the normal array of audio-visual paraphernalia, the training van carries a Bell and Howell Apple, a TI 99/4, and a TRS-80. When a school system decides to investigate the benefits of computers in the classroom, teachers and administrators may request copies of "A Guide to Microcomputers" and "A Guide to Instructional Microcomputer Software" both compiled by Ms. Douglas and Gary Neights, Coordinator of the Instructional Materials Service Programs in the state. These booklets provide information on computer literacy and guidance in developing purchase criteria.

The district may also request one of the "In-Service Programs" offered by Shirley in the van. Her introductory half-day course is entitled "Microcomputers," and assists participants in developing the criteria and purchase specifications mentioned in the booklets. It offers hands-on experience as well as discussion and demonstrations.

In "A Guide to Microcomputers," prospective purchasers are urged to identify the specific uses to which the new computer will be put. A list of "Projected Uses" (Figure 1) serves as a guide during this initial phase of the selection process.

Once the uses have been defined, the people who will use the machine are asked to consider such criteria as cost, flexibility, mainframe interface, keyboard layout, execution and loading speed, memory capability, system expansion and many others. A tally sheet lists 17 of these criteria; the prospective user assigns a number between one and ten to indicate how important he or she considers each feature. If the school plans to use the computer in more than one area, or for



PROJECTED USES

I. Initially determine curriculum areas of use and possible other uses.

Subject Areas

_____ math
_____ science
_____ social studies
_____ health
_____ industrial arts
_____ home economics
_____ reading
_____ language arts
_____ foreign language
_____ business education
_____ physical education
_____ other ____________________

Other Uses

_____ guidance
_____ library science
_____ media
_____ management
_____ computer literacy
_____ other ____________________
____

II. If the microcomputer is to be used as an instructional tool (i.e.: math, art, music, etc.) the next step is to determine the microcomputer's utilization by the classroom teacher and the student.

Classroom Teacher Application

_____ curriculum (subject area teaching)
_____ computer operation and programming
_____ computer literacy instruction
_____ testing
_____ classroom management (teacher's record keeping)
_____ other ____________________

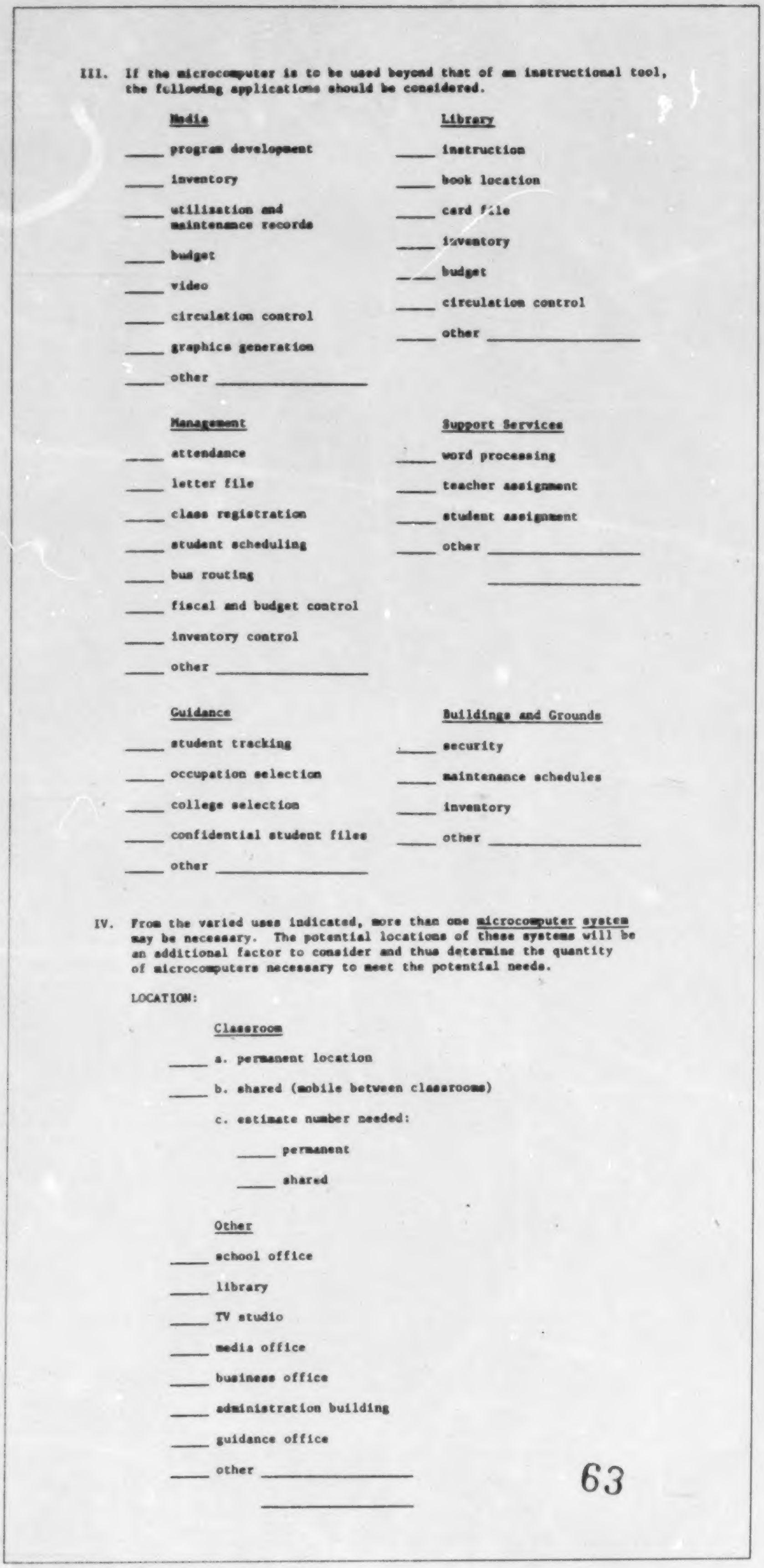
Student Application

_____ discovery learning
_____ problem resolution
_____ graphics development
_____ musical exploration
_____ computer programming
_____ computer awareness
_____ counseling and guidance
_____ other ____________________

Figure 1.

Charts and questions reprinted from "A Guide to Microcomputers" and "A Guide to Instructional Microcomputer Software" with permission of the authors, representing the Pennsylvania Department of Education.

From *Creative Computing,* Volume 7, Number 3, pages 106–108, 110, 112. Copyright © 1981 by Creative Computing. All rights reserved. Reprinted with permission.



III. If the microcomputer is to be used beyond that of an instructional tool, the following applications should be considered.

Media

____ program development
____ inventory
____ utilization and maintenance records
____ budget
____ video
____ circulation control
____ graphics generation
____ other ____________

Library

____ instruction
____ book location
____ card file
____ inventory
____ budget
____ circulation control
____ other ____________

Management

____ attendance
____ letter file
____ class registration
____ student scheduling
____ bus routing
____ fiscal and budget control
____ inventory control
____ other ____________

Support Services

____ word processing
____ teacher assignment
____ student assignment
____ other ____________
____

Guidance

____ student tracking
____ occupation selection
____ college selection
____ confidential student files
____ other ____________

Buildings and Grounds

____ security
____ maintenance schedules
____ inventory
____ other ____________

IV. From the varied uses indicated, more than one microcomputer system may be necessary. The potential locations of these systems will be an additional factor to consider and thus determine the quantity of microcomputers necessary to meet the potential needs.

LOCATION:

Classroom

____ a. permanent location
____ b. shared (mobile between classrooms)
c. estimate number needed:
  ____ permanent
  ____ shared

Other

____ school office
____ library
____ TV studio
____ media office
____ business office
____ administration building
____ guidance office
____ other ____________
____

more than one application, all prospective users should assign "importance factors" based on their individual needs.

When importance factors have been assigned, individual computers can be considered. Based on promotional literature, information provided by sales people, or experience with one of the machines in the van, a "Comment Sheet" can be completed. Figure 2 shows a comment sheet for the mythical Spacetron 007.

On the final tally sheet, users are asked to consider again the 17 criteria and assign a value between one and five for the computer in question. "If the cost is high, give it a 1. If it is easily movable and compact then flexibility may be a 5." These "single rankings" are then filled in on the tally sheet and multiplied by the importance factors to produce a rating for each criterion. (See Figure 3.) When the ratings are totalled, the user has a number which can be compared to the totals for other machines or for the same machine in other departments. While obviously not infallible, this technique provides a method of quantifying what might otherwise be left to someone's gut feeling. The instructions conclude by cautioning: "Color of the case, unnecessary extras or the salesperson's personality should not affect the choice."

Software Selection

Once the computer system has been selected, the problem of which software to purchase may be overwhelming. Very few manufacturers are willing to send sample programs, and it may be difficult to find educational software in the local computer store. Here comes the van again.

Shirley purchases single copies of educational programs from many different vendors and makes them available in the van to educators who want to consider them for adoption. If a teacher decides to adopt a given program or package, it must be purchased through his or her school district. Software carried in the van may not be copied.

The software evaluation form (Figure 4), found in "A Guide to Instructional Microcomputer Software," is designed to assist in the selection of software. First an instructional objective must be stated. Then the objective is compared to the features of the program with regard to grade level, validation, correlation with text, instructional strategies and instructional design features. It also includes room for a description of the program and an overall evaluation.

After hardware and software are chosen, teachers may avail themselves of a second In-service program which deals with "Microcomputer Classroom Applications." During the full-day course, participants learn to operate the computer, run programs, and even write a short program in Basic.

Van, continued...

Questions to Ask About a Computer System

1. **Cost:** This is a factor to be considered. Micros generally range from $500 to $3,000 to establish a system. This cost is to be with all peripherals needed to operate.
2. **Flexibility:** (Size, portability, cords and modules, environment) Depending upon needs can the unit be readily moved? Is it necessary for the unit to be moved? Is it sturdy and reliable to survive moving around? Has it been tested for durability? How much does it weigh? Is it necessary to be near an outlet or telephone lines? Do the learners have to be brought to the unit or does the unit have to be taken to the learners? Can the unit be accessed other than being right at the microcomputer itself? How many cords necessary to operate the micro? Is there a need for special environmental controls, i.e., temperature, humidity, dust? (Protection from exterior electrical interference, i.e., other computers, static charges, another electromagnetic field)
3. **Mainframe Interface:** Does the unit have the ability to interface with available mainframe computers to function as a smart terminal?
4. **Keyboard Layout:** Most micros come with a standard typewriter layout. If the unit does not, will it fit your needs? Does it have a calculator layout on it? Is a calculator layout necessary or can the standard typewriter numbers fulfill the calculation needs?
5. **Additional Ports:** Can other peripheral devices be connected to the unit, i.e., printers, plotters, phone lines, disks, etc.? Are there sufficient ports to substantiate your operational needs? Do these ports use memory (RAM) that would otherwise be available?
6. **Execution time and loading speed:** How long does it take the microcomputer to execute an operation? How fast can information be loaded into the unit? Is the execution time, problem, operation or loading of a program too long for student-attention spans?
7. **Memory capability:** How much ROM memory is the unit capable of? How much RAM memory can be taped? RAM is found in varying forms. If strictly for running of prepackaged materials then usually 16K will suffice but generally self-generated programs will take more bytes of memory (RAM).
8. **System Expansion:** Can the system be expanded easily? What are the limits of the expansion? What peripherals are available? Are peripherals needed? With the current state of advances new items for purchase are always being developed, such as light pens, graphics, tablets, voice synthesizers, etc. Maybe even keys for the blind or some other new advances are in the making as this pamphlet goes to publication.
9. **Editing:** Can editing take place immediately as mistakes occur? Is editing simple? After the program is completed can editing be done? Can changes in the program to suit needs be done? Will the unit identify specific program errors?
10. **Input and Output Devices:** As specified for purchase what input and output devices are included in the package, i.e., cassette, disk, TV monitor, printer, plotter, graphics tablet, light pen, voice synthesizer?
11. **Software:** Are there sufficient manuals, reference and program material available to support the microcomputer? Are there programs suited to the user's needs? Have outside companies (other than the original designer) make additional software programs? Is there enough software available to fulfill needs? Can programs be made to fulfill the user's needs? Have the programs been validated (field tested with students)? Cost of prepared programs? Ease of self-generated programs?
12. **Graphics/Characters:** Is the unit capable of low or high resolution graphics? How many characters per line are available on the micro? How many lines on the CRT are visible? What is the screen size? Graphics tablet? Light pen?
13. **Color:** Is color necessary for your operations? If color is necessary, does the CRT monitor have to be a special monitor?
14. **Voice command and voice generation:** Does the unit have voice synthesizers, to generate voice, does it have or can it be adapted to accept voice commands?
15. **Music Generation:** Is there music capability? Does it have an internal speaker or separate speaker system for sound?
16. **Servicing:** What are the warranties available? Can the unit be serviced at its home base? Is on-site servicing necessary? Can local technicians make necessary repairs? For additional cost will the unit be updated for the next year as new developments are made? Length of time for service including transportation to and from service facility? Service cost including transportation?
17. **User Training:** Will vendors provide on-site user training? At what cost? How many hours? For how many people?

Instructional Materials Service Technicians learn to service and repair the Bell and Howell Apple.

When evaluating a microcomputer, use the comment sheet. Write notes about that particular model and brand. To better evaluate the computer, use the criterion questions 1-16 on pages 10 and 11. The notes on the comment sheet will be translated into numbers on the tally sheet.

COMMENT SHEET

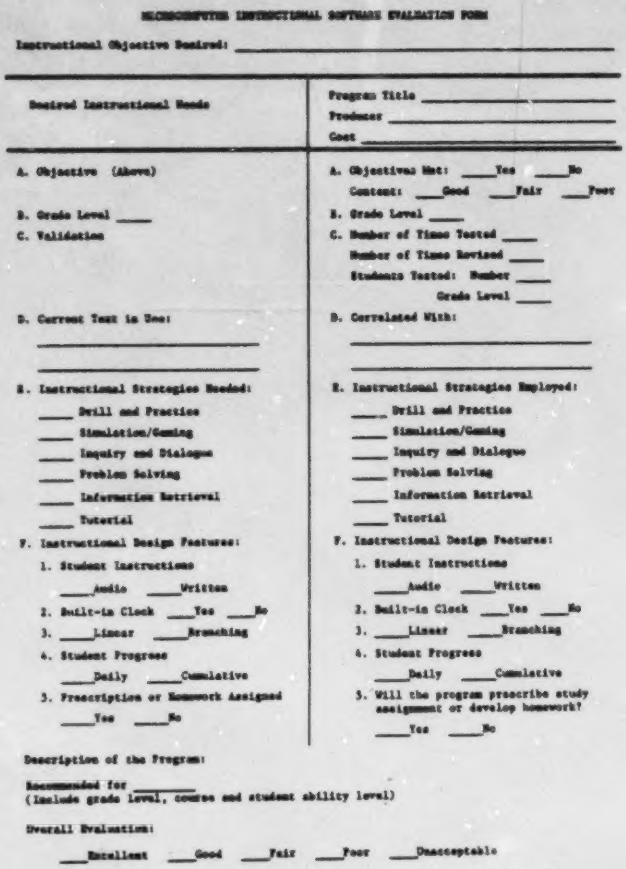
CATEGORY	BRAND Spacetron MODEL 007	BRAND MODEL
1. COST--TOTAL	995 - 16K 998 - 32K	
2. FLEXIBILITY	[illegible]	
3. MAINFRAME INTERFACE	none	
4. KEYBOARD LAYOUT	Standard w/calculator	
5. ADDITIONAL PORTS	8ports uses RAM	
6. EXECUTION TIME & LOADING SPEED	L.S. - 5sec 1.3sec E.T.	
7. MEMORY CAPABILITY		
8. SYSTEM EXPANSION	48k	
9. EDITING	none	
10. INPUT AND OUTPUT DEVICES	printer, plotter; disk,	
11. SOFTWARE	8 programs math manuals	
12. GRAPHICS/CHARACTERS	hi resolution	
13. COLOR	14 color stand. T.V. monitor	
14. VOICE COMMAND AND VOICE GENERATION	no	
15. MUSIC GENERATION	no	
16. SERVICING	none	
17. USER TRAINING	none	
18. TOTALS	N/A	

On the tally sheet using a scale ranking (1-5) assign a rank value for each category from the comment sheet. If the cost is high give it a 1. If it is easily movable and compact then flexibility may be a 5. If no service is available category (16) is assigned a 0, etc. When all categories have been assigned a scale ranking, multiply them by the importance factor you originally assigned. The end results are added together. The highest total (18) will indicate the microcomputer best suited for your uses and needs. Be as sincere as possible to get an unbiased evaluation. Color of the case, unnecessary extras or the salespersons personality should not affect the choice.

TALLY SHEET

CATEGORY	IMPORTANCE FACTORS	BRAND SPACETRON SCALE RANKING	RATING
1. COST--TOTAL	5	1	5
2. FLEXIBILITY	5	5	25
3. MAINFRAME INTERFACE	10	0	0
4. KEYBOARD LAYOUT	3	5	15
5. ADDITIONAL PORTS	10	4	40
6. EXECUTION TIME & LOADING SPEED	10	3	30
7. MEMORY CAPABILITY	10	2	20
8. SYSTEM EXPANSION	8	5	40
9. EDITING	10	0	0
10. INPUT AND OUTPUT DEVICES	9	3	27
11. SOFTWARE	10	2	20
12. GRAPHICS/CHARACTERS	8	5	40
13. COLOR	5	5	25
14. VOICE COMMAND AND VOICE GENERATION	5	0	0
15. MUSIC GENERATION	3	0	0
16. SERVICING	10	0	0
17. USER TRAINING	10	0	0
18. TOTALS	—	—	297

Figure 3.



MICROCOMPUTER INSTRUCTIONAL SOFTWARE EVALUATION FORM

Instructional Objective Desired: ______________________________

Desired Instructional Needs	Program Title ______________ Producer ______________ Cost ______________
A. Objective (Above)	A. Objectives Met: ____Yes ____No Content: ___Good ___Fair ___Poor
B. Grade Level ____	B. Grade Level ____
C. Validation	C. Number of Times Tested ____ Number of Times Revised ____ Students Tested: Number ____ Grade Level ____
D. Current Text in Use: ______________ ______________	D. Correlated With: ______________ ______________
E. Instructional Strategies Needed: ____ Drill and Practice ____ Simulation/Gaming ____ Inquiry and Dialogue ____ Problem Solving ____ Information Retrieval ____ Tutorial	E. Instructional Strategies Employed: ____ Drill and Practice ____ Simulation/Gaming ____ Inquiry and Dialogue ____ Problem Solving ____ Information Retrieval ____ Tutorial
F. Instructional Design Features: 1. Student Instructions ____Audio ____Written 2. Built-in Clock ___Yes ___No 3. ____Linear ____Branching 4. Student Progress ____Daily ____Cumulative 5. Prescription or Homework Assigned ____Yes ____No	F. Instructional Design Features: 1. Student Instructions ____Audio ____Written 2. Built-in Clock ___Yes ___No 3. ____Linear ____Branching 4. Student Progress ____Daily ____Cumulative 5. Will the program prescribe study assignment or develop homework? ____Yes ____No

Description of the Program:

Recommended for ________
(Include grade level, course and student ability level)

Overall Evaluation:

___Excellent ___Good ___Fair ___Poor ___Unacceptable

Figure 4.

Other Services

The van staff has taken special care to reach administrators, since they are most frequently in decision-making positions. The Microcomputer Administrators Days Workshops, in which administrators get an overview of small computers in education as well as specific information on hardware specifications and evaluation and software evaluation, have reached over 50% of the school district administrators in the state.

Perhaps the most innovative program in this innovative program was the repair seminar conducted in March 1980, in which Bell and Howell Apple personnel trained approximately 20 of the state's Instructional Materials Service Technicians to repair their computer. Negotiations are underway with several other computer manufacturers to provide similar sessions.

Shirley points out the participants in these and other courses and services provided by the van are expected to be "multipliers." Key personnel are trained to train others in the school or district. The van enables one person and a few pieces of audio-visual and computer equipment to serve many people and many school districts all over the state. ☐

6 The Human Perspective

In the event that the reader has become convinced that computer technology should be embraced at any cost even at the expense of substituting machines for human resources, the following two articles have been included. Both serve to remind us of the value of human relationships.

In *Some Thoughts on Computers and Greatness in Teaching,* Thomas A. Dwyer comments, "...we would be suspicious of an overoptimistic prognosis for the future of computing in education if it didn't also address the importance of developing greatness in teachers." Because of computer technology, the teacher is freed to become a codiscoverer of truth with the student. Dwyer views computing as a backup system that can induce greatness in teaching but requires a new approach to teacher education. The author calls for a return to greatness in teaching.

The Hacker Papers are a collection of communications among college students and other computer specialists. As their electronic mail unfolds, the reader gains insight into some of the problems associated with being a Hacker or a computer specialist whose interests are limited to interactions with computers and other computer specialists.

This article should be required reading especially for parents who are considering purchasing a personal computer. Forewarned is forearmed. There is little question that home computers should probably be made available whenever possible but parents and students themselves should be aware of the potential consuming attraction inherent in computer experiences. There is a need to make a conscious effort to encourage or engage in activities that will insure a well-rounded individual.

Some Thoughts on Computers and Greatness in Teaching

THOMAS A. DWYER
Project Solo
University of Pittsburgh
Pittsburgh, Pennsylvania 15260

In a paper entitled "The Computer and the Fourth Revolution," Molnar (1973) gave a panoramic view of recent efforts to harness the potential of the computer for education. He concluded that "computing is so compelling a tool that it cannot be stopped." Our own experiences, both at Project Solo and through numerous contacts with others, confirm this conclusion, and in fact suggest an even stronger one. We have found that computing, placed in the hands of well-supported teachers and students, can be an agent for catalyzing educational excitement and growth of a kind that is without precedent. We believe that there has simply been no other tool like it in the history of education.

Great Teaching in the Mass Education System

Our enthusiasm is tempered by a growing respect for the importance of human relationships in education. In particular, we would be suspicious of an over-optimistic prognosis for the future of computing in education if it didn't also address the importance of developing greatness in teachers.

Developers of computer-related instructional systems seldom make explicitly negative statements about the role of teachers. However there have been unfortunate implications that computers are important because they can make education "teacher-proof." This is not only an incorrect attitude; it cuts off the real power of the computing tool, namely the potential to multiply the effectiveness of the users of that tool. The most important users are not research and development people, but "teacher/student people." Put simply, I believe that computers in education are revolutionary because they make possible great teaching in a system dedicated to mass education. They make this possible by supporting the awesome mystery of person-to-person educational influence, not by replacing it.

To see the consequences of this last statement for teacher education in the computer age, let's back up a bit. It is a truism that there have been, and always will be, inspirational teachers. The know-how with which they have practiced their art has remained pretty much of an untapped resource, however. The classical "pipeline" model of educational R&D (educational product shipped to end users *after* development) makes no attempt to tap his resource. How, it would be argued, could as ill-defined a concept as "great teaching" ever be encapsulated and shared in a mass education system? More to the point, how could a word like *greatness* ever characterize the efforts of the majority of teachers who are (by definition) "average" educators?

It has been tempting to respond to these questions by proposing that the art of the great educator be captured and disseminated through the medium of "computer assisted instruction." One early worker in CAI went so far as to predict that "millions of school children will have access to what Philip of Macedon's son, Alexander, enjoyed as a royal prerogative: the personal services of a tutor as well informed and responsive as Aristotle" (Suppes, 1966). CAI is also based on a pipeline model. The years have verified that what comes out of the pipe, even when a computer is at the other end, bears little resemblance to the charisma of an Aristotle. CAI has its virtues, but reproducing the mystery of person-to-person teaching is not one of them.

Solo-Mode Computing

There is another use of computer-related technology, however, that is triggering originality and depth in student work, as well as such personal qualities as enthusiasm and zest, particularly in the area of math education. It is doing this, not so much by reproducing pedagogical greatness, but rather by *inducing* such greatness.[*] It is a use of technology that works its magic by freeing teacher/student groups from the restricted format of drill in facts and formulas. Teacher and student become, instead, codiscoverers of truths. The methods they learn to use, and the results they obtain, display a freshness that suggests that they have personally discovered a secret that transcends the art of any one great teacher. Their secret, put simply, is to use computer technology to build an environment in which learning mathematics is both natural and exciting. Because their work is predicated on the belief that the student alone must (ultimately) control what is done with the technology, such use has been called "solo-mode" computing (Dwyer, 1974). Because the technology provides a world of experiences to be lived in by both teacher and student, the resulting environment has been affectionately dubbed "mathland" (Papert, 1972).

On the negative side, it should be clear that environmental solo learning, devoid of guidance and support, can lead to harm as well as good. A person who lives in the wrong part of France can learn bad French; South American babies who learn to swim by falling off the family houseraft don't always survive. Like most good ideas, environmental solo learning needs to be coupled with other factors before it reaches its full potential.

I believe that the strength of teacher and student-controlled computing is that it automatically provides many of these factors; because the students are responsible for what the computer does, there is a kind of self-validation in student-controlled computing. Of course such computing is an exceptionally rich source of solo-mode environmental learning.

We have recently been experimenting[*] with all kinds of other rich environments, and it is becoming increasingly clear to us that the general-purpose computer belongs at the heart of such environments (sometimes just conceptually). Solo-mode computing, properly supported, is undoubtedly the component par excellence for an experiential educational milieu.

---

[*] In a subtle way, clever technology can also "reproduce" some of the talent of the inventor (e.g. Papert's Turtle), and in this way transmit certain aspects of great teaching.

[*] These experiments are part of a project called "Soloworks" which is concerned with developing computer-based math lab environments. This work is supported in part by NSF grant EC-38063. A project newsletter is available from the author.

The thing that pleases us most about this discovery is that it supports a "romantic" view of education that is also eminently practical. In fact it is downright hard-nosed in its relation to reality. This is because it makes contact with the enduring reality of education today (and for a long time to come), with the fundamental "people-unit" of a large educational system, namely the teacher/student group.

Computers as Backup Systems

A realistic view of education admits that human teaching can be uneven and sometimes very sloppy, but it also understands that the relationships involved between students and the teacher far transcend the "logic" of even the fanciest instructional flowchart. A realistic view sees students and teachers as deserving of all the help and encouragement they can get as they wrestle with some very tough problems. They are people working with an outdated set of tools in a system whose priorities do not recognize the importance of educationally oriented support systems. There is an innate "genius" to teacher/student groups, but the flourishing of this genius is next to impossible without sophisticated backup. Bruner (1970) stated the need emphatically:

> . . . in 1965 I took a leave of absence from Harvard and taught fifth grade for three months. . . . I taught every day until 1:15 and came back to my office exhausted. Like the other teachers I had to prepare most everything myself. Now I know that after you teach for a while, you can't expend energy that hard and lay your ego that constantly on the line. How can a teacher cope without some kind of supporting backup?

Thus, supporting backup is the essence of reality; when it is backup that also induces unique greatness in teacher/student work (as does locally controlled computing), then it deserves being called a "compelling" educational phenomenon.

Teacher Education—Preparation for "Greatness"

A consequence of viewing computing as a backup system that can induce greatness in teaching is that one gets a new view of teacher education. One understands, for example, that it is very important to prepare teachers to use such "supportive backup" systems *creatively*, and to help them think about the importance of getting their students to do the same. I use the word *creatively* to suggest that teachers should be educated with the view that they are life-long researchers, not technicians who implement other people's ideas. Another difference would be to immerse teachers in very rich technological environments as part of their education, again with the objective of making them fluent at extending (rather than just imitating) ways of using similar supporting systems.

The overall goal of teacher education should be to give teachers status based on expertise—to help them master the art of molding themselves and their students into very strong personal units within the large impersonal systems that arise in mass education. Of course such strength will not come easily; it will have to be earned. One of the real contributions of a locally controlled computer system is to facilitate the kind of accomplishment that will earn respect and status for both teachers and students. Such a system can also have a protective function, insulating teacher/student groups from the erosive forces of noneducational concerns found in a big school system. Any teacher who has worked in a school with good departmental physical facilities will understand the significance of this remark.

Conclusion

A good way to summarize these remarks is to say that I believe computer technology has brought us the capability of making a very fundamental change in education. The change is a return to greatness in teaching. It is a change that eschews the use of technology to automate pedagogy. On the contrary, it's a view that starts out by recognizing the extraordinary power of human relationships. It believes that teachers are important because they are the proper focus of such relationships in education. It sees technology as a compelling force in education because it has the power to melt away the real-world constraints that manacle even the best of teachers. In a very real sense, it is a view that says we are now able to back off from the frightening dead-end of school systems without belief in their own power to teach, and to support the methods that we knew to be right all along.

References

Bad education—A conversation with Jerome Bruner. *Psychology Today*, 1970.

Dwyer, T. Heuristic strategies for using computers to enrich education. *International Journal of Man-Machine Studies*, 1974, *6*, 137-154.

Molnar, A. *The computer and the fourth revolution*. Paper presented at the Association for Educational Data Systems conference, New Orleans, 1973.

Papert, S. *The computer as super-toy*. Paper presented at SJCC, 1972.

Suppes, P. The uses of computers in education. *Scientific American*, 1966, *215*(3), 206-220.

THE HACKER PAPERS

They were among the brightest computer-science students at Stanford. They spent their days and nights in the concrete-and-glass computer center playing harmless games and mock-vicious "wheel wars" on the terminals. Then, one night, a hacker (computer addict) whose code name was Gandalf sat down and tapped out a *cri de coeur*: in their single-mindedness, he and his fellow hackers were losing touch with the human race. Quickly the issue was joined. The complete printout follows. (For a glossary of hacker terminology, turn to page 79 .)

From *Psychology Today Magazine*, August 1980, Volume 14, Number 3, pages 62–69. Copyright © 1980 by Ziff-Davis Publishing Company. Reprinted with permission.

STANFORD UNIVERSITY
LOW OVERHEAD TIME-SHARING SYSTEM (LOTS)
FROM: G. GANDALF
TO: BULLETIN BOARD
SUBJECT: ESSAY ON HACKING

Dedicated to all my friends at LOTS who will live their lives in an alien culture surrounded by humanity, and to Ernest, who was too human for it.

As much as an essay, this is a story. It is a true story of people paying $9,000 a year to lose elements of their humanity. It is a story of both seen and unseen games. It is a story of the breaking of wills and of people. It is a story of addictions, and of misplaced values. In a large part, it is my story.

There is no one field in particular in academia that has a monopoly on production of single-interest people, and this practice can exist almost anywhere. There is the political-power seeker, all-consumed by climbing up the bureaucratic rungs. There is the stereotyped premed, ignoring all but his MCAT scores. There is the compulsive artist or writer, forever lost in his work. Narrowness is widespread. But there is one field that seems to be more consistent in this practice. This essay, rooted in personal and painful experience, is about the people in computer science.

In the middle of Stanford University there is a large concrete-and-glass building filled with computer terminals. When one enters this building through the glass doors, one steps into a different culture. Fifty people stare at terminal screens. Fifty faces connected to 50 bodies, connected to 50 sets of fingers that pound on 50 keyboards ultimately linked to a computer. If you go farther inside, you can discover the true addicts: the members of the Establishment. These are the people who spend their lives with computers and fellow "hackers." These are the members of a subculture so foreign to most outsiders that it not only walls itself off but is walled off, in turn, by those who cannot understand it. The wall is built from both sides at once.

These people deserve a description. In very few ways do they seem average. First, they are all bright, so bright, in fact, that they experienced social problems even before they became interested in computers. Second, they are self-contained. Their entire social existence usually centers around one another. Very, very few remain close to their families. Very, very few associate much with anyone who is not at least partially a member of the hacking group. While they do sometimes enjoy entertainment unrelated to their field, it is almost always with fellow hackers. Third, all aspects of their existence reinforce one another. They go to school in order to learn about computers, they work at jobs in programming and computer maintenance, and they lead their social lives with hackers. Academically, socially, and in the world of cash, computers are the focus of their existence.

The hacker will probably not strongly disagree with what has been said so far. But he will ask the question, "So what?" The answer is: in creating a subculture and isolating it, we are destroying the chance that computers might be used wisely as an integral part of our society. We are precluding the human values so necessary for the wise application of this technological achievement. The most brilliant young minds at our top universities are learning how to play with multimillion-dollar toys first, and how to utilize them constructively second.

Even if we ignore the costs to society as a whole, we have to look at the costs to the people involved. The computer is a modifier of personalities. It is highly addictive. People who gain this addiction for a period of several months tend never to give it up. And the symptoms are very sad.

The first thing to go is other academic interests. Basically what occurs is that the hackers' motivation to challenge themselves in any field not directly linked to computers gradually disintegrates. On the level of grades, straight-A students tacitly accept C's in noncomputer courses. On the level of actual learning, the same students shut off outside subjects even more completely than their grades would indicate. This is common in many areas of specialization, but nothing compares with the incredible consumption of computer-science students for computer-science courses, and their nonchalant attitude toward every other class.

The second thing to go is a normal living pattern. Eating and sleeping are completely rearranged to fit the addiction. The typical hacker thinks nothing of eating one meal a day and subsisting on junk food, or of sleeping from 4 a.m. to noon almost every day of the week. Families are soon disregarded, to an extent uncommon even when one considers the separation that generally occurs in college. It is simply that the parents of hackers are ignorant of the subculture and cannot relate.

The third thing to go is a balanced social life. The hackers' narrowness and strange schedule simply compound the social problems they experienced before hackerdom. Soon, no one except a hacker has the capability to understand other hackers. No one except a hacker will go out with other hackers. No one except a hacker can talk to another hacker.

The fourth and final thing that happens is also the saddest. The personality of the hacker shifts, in order to permanently adjust to the new social conditions. Emotions always have hurt before, so they are effectively isolated. Relations with nonhackers become strained, so why force the effort? It is so much easier just to accept social rejection and isolation, and to do it with a spirit of camaraderie that's shared by the rest of the subculture.

An essay should make an attempt to resolve the problem it points out. In this case, the pointing may be enough, or at least enough to do whatever can be done. I know from personal experience what a trauma it can be. I was one of the top 10 among several thousand LOTS users last spring for the amount of time I spent here. I have watched people close to me undergo the transformation. I narrowly escaped it.

The tragedy is that I am so involved in piecing my personality and social life back together that I think I have learned very little about how to prevent this from happening in the first place. I am lucky. I will go on to some sort of a balanced life (although my hacker friends will laugh at me, since, to them, my involvement was never serious enough to make me one of them). Weak-willed people, people with unstable social lives, people in formative stages of their lives, should not become involved in computer science. It should be left until they are truly able to make decisions and be aware of all the consequences. Computers are most often used by people who start when they are immature. This is what causes the single-minded addiction. This is what takes some of the brightest and most capable minds in college today and turns them to narrowness.

FROM: E. ERNEST
TO: G. GANDALF
SUBJECT: REPLY TO "ESSAY ON HACKING"

As the human to whom Gandalf's essay was in part dedicated, I feel I have a responsibility to reply. Obviously, his essay is going to provoke a certain amount of defensiveness from the hackers; on the other hand, he has made a number of points which are indeed very valid.

I became rather disillusioned with LOTS last spring because I found that the attitudes of the people whom I had made my friends were becoming more and more machine-oriented and less and less human-oriented. I began to spend less time here, and see the people less often. It was partly the so-called narrowness to which Gandalf referred, but it was also a simple lack of genuine human caring on the part of these people that I left.

I'm not concerned about their social lives: physics majors, premeds, law students, the regulars at Aunt Minnie's church social—all behave in roughly the same way. Nor am I concerned about their grades, for the same reason that I'm not concerned about anybody's grades, including my own; if they choose to get hassled over grades, they'll do so, and they'll make them if they want them. I AM concerned about the narrow-mindedness that comes from losing interest in such outside things as the draft or the social sciences, or all the various and sundry things that Stanford has to offer. This place should be a broadening experience; to fail to take advantage of that is a waste of $9,000. Furthermore, narrow-mindedness and xenophobia have started more wars than any other cause.

More than anything else, I'm concerned about the hacker's shortage of respect for others and a lack of sympathy for their problems, because he considers the problem of whether or not he wants a HRROI or a STI in a certain place to be more fun than people are. (If I get a lot of mail telling me that that's a ridiculous example and no one could possibly mistake a HRROI for a STI, it will only prove my point.) By and large, the hackers are consultants, and as consultants they owe it to the users to be patient, pleasant, and, above all, sympathetic. A large number of people in introductory computer-science courses are not cut out for computer work: the thought processes it requires have long ago been squashed by indoctrination and training. They are confused, frustrated, and upset. We do the best we can for them by teaching them PASCAL and giving them an interactive system with a forgiving exec which holds their hands at every turn. But there is nothing like human comfort to make people feel better about it.

Gandalf's problem was in attributing these faults to the computer. The machine itself is not responsible for it: the attitudes of the others toward the machine are to blame. After having spent a summer doing data processing, I know that not all computers have this fascination. Look at J.Q., Vic, Sean, or Karl. They don't demonstrate these attitudes. They know that, fascinating as it may be, the computer is a tool and not a toy. The fault lies in the attitude of the individual.

What is to be done? Nothing is to be done. The hackers know who they are, running around and having "wheel wars" late at night, yakking and yakking while someone stands and waits for them in front of the consultant's terminal. We may only hope that they grow out of it. Gandalf is wrong. It isn't a permanent alteration in their character—it is, hopefully, a short-lived immaturity of people on their way to becoming programmers and computer scientists.

FROM: K. KANEF
TO: E. ERNEST
SUBJECT: IN REPLY TO REPLY TXT

In your opinion, the two problems that LOTS hackers have developed are a loss of touch with their humanity and an immature attitude about the purpose of the computer. Until recently, I would have disagreed with the first assertion. I'm no longer so sure that it's not true.

I have even stronger feelings about the other claim that you make about hackers: that they have an immature view of the purpose of the computer. You assume that childishness is a thing to be avoided. You encourage people to throw away the things that are so hard to retain in growing up: the sense of wonder in new wonderful things, the ability to PLAY. When I compare my attitude to that of Vic and the others you mention as having a more mature view of the computing world, I find that I like mine better. I don't want to be like them! I see the computer as a toy as well as a tool. And as I pointed out to Gandalf last week, this attitude is not limited to hackers. There were many people in my dorm last year whose only knowledge of computers was how to play games and send mail, and they enjoyed using the computer as much as I did. I was one of the few people in that group who was interested enough in the field of computer programming to also take an intro course in PASCAL.

I hope that I will always be as fascinated by my work as I am now. Unfortunately, I know that this is not likely to happen; all too often people become bored with their professions. You're right in saying that the childishness of LOTS hackers is short-lived—and isn't it a shame?

FROM: E. ERNEST
TO: K. KANEF
SUBJECT: THE COMPUTER AS TOY

I am impressed by your skeptical intelligence, and I appreciate your reply. I am sorry that you became aware of the diminution of simple humanitarian caring on the part of hackers the hard way, but pleased that you know of its existence.

As regards play, you present a very convincing argument, but I would like to add a qualifier.

To me, as one matures, the highest forms of play cease to be consumptive and begin to be productive. Sure, we all need to play, and there is nothing wrong with consumptive play in an appropriate setting and appropriate circumstances. Middle-aged adults enjoy Great America, and this to me seems to be a perfectly healthy and appropriate form of consumptive play. Nonetheless, I don't believe it is the highest form of play that they should be capable of.

Computer games are tolerable and serve their purpose; even they help to acquaint new users with the computer and to show that it is not a thing to be frightened of or confused by. But programs that type "You lose" when you type "sigh" to the exec don't serve any productive purpose at all; they are vaguely amusing and they take up disk space, and an entire page for six lines of code, at that. Likewise, the wheel war is neither educational nor productive; it is nothing more than consumptive play, and it can be annoying.

FROM: M. MRC
TO: G. GANDALF, E. ERNEST
SUBJECT: NONE GIVEN

Most of the behavior you described is more or less typical of an undergraduate-aged person and less typical of hackers as they get older. Rather than demeaning it, we should simply recognize it as normal for somebody that age. In a high school kid, it is actually a sign of maturity to be at that stage so quickly. In a graduate student, it's regressive.

Don't be misled by those who postulate the "compulsive" (and hence parasitic) programmer versus the "professional" programmer—I know of no programmer who does not consider the computer as a toy first. You really do have to enjoy this work to be good at it. I know of no hacker who does not consider the computer as a toy first. What changes are the ways that s/he plays with the toy. There isn't all that much difference, in terms of the pleasure derived, between writing a new display hack or a clever access-control program, except that the former is something derided as being "hacking" and the latter is "responsible programming."

I guess my point is that it's all a part of the irresponsibility that college undergraduates normally have, and it shouldn't be begrudged them; it's the last chance they are ever going to have in their lives to be that free.

FROM: A. ANONYMOUS
TO: E. ERNEST, K. KANEF
CC: G. GANDALF
SUBJECT: THE HACKER CHRONICLES

We are dealing with an infinitely malleable tool. People who choose to develop and use that tool, whether for work, play, or both, have that choice and cannot be denied it. A person who chooses to be a musician must devote hours and hours to gain adequate expertise. But would you consider the computer hacker any less creative than such a person? I certainly wouldn't. The computer serves not only as a workhorse, but also as an easel for exercising one's creative abilities. Therefore, in my opinion, the hacker has not limited himself at all. Rather, he has expanded his intellectual horizon because now he has the infinite tool.

As for the charge that it disrupts one's social life, I would tend to agree with this to a point. But it depends on how controlled the individual is. At any time, he can withdraw to a more normal schedule. Why doesn't he? The reason is obvious. The infinite tool that knows few boundaries is accessible to a much higher degree, and thus he can devote more time to it. Why is this wrong? I think it is definitely a bonus, since the usual restraints of 9-to-5 are eliminated and the person is allowed to expand beyond boundaries to do what he wants.

Now we come to the human versus the machine factor. Gandalf stresses the necessity of human interaction and the inherent evil of the machine. Would you stress the evil of the instruments in an orchestra, or the instruments in a laboratory, or the typewriter of an author? All of these occupations demand extraordinary amounts of time for excellence. But I see no greater human interaction in these fields than in computers. I feel that people who disparage computers for a seemingly decreased human interaction are not at all familiar with the true import of the computer. Not only is it the infinite tool, it is also an extremely fluid medium of communication.

FROM: K. KUDDLES
TO: G. GANDALF
SUBJECT: HACKING AND THE REAL WORLD

I don't want to turn this into a personal defense or anything like that, but as I've said before, people have always found me strange. I have always worn my hair long because I felt it was comfortable for me. I prefer clothes that are comfortable and perhaps a bit scroungy. For this, and being human, and getting excited, I have gotten pretty bashed. As a matter of personal history: it was nice for me to get away from a smog-hole of a city, where the mayor said aloud on TV once that the "black problem" existed because the "blacks weren't disciplined properly." I spent time in an ad agency where people were not only truncated and anesthetized, but positively deformed in personality. They viewed people as real, honest-to-goodness objects of $$ and/or pieces of meat. They'd promise you anything, but split with your portfolio.

Working with computers has been the only thing tolerable I have done in quite a while. It does have its drawbacks, and you can get into just doing that and not trying anything else. Part of it derives, I think, from a strange version of the American work ethic. The thing I find sad is when hacking comes before people; and it does sometimes for me, usually where $$$ and all this financial-survival crap is concerned. But it is saddest when someone will not take time out from hacking to be with people at all. I mean when people get angry because someone sends them a message asking them what they are up to, and the other person gets all freaked-out and sends nasty notes back.

The problem is that the social structure of LOTS rewards people who hack well but don't interface well with people. If you look at who gets most rewarded at LOTS and for what, you find it is the people who do programs and little maintenance tasks. Very seldom do people who get along with others and relate well to them get highly rewarded.

FROM: G. GANDALF
TO: A. ANONYMOUS
SUBJECT: A REPLY

You base much of your argument on comparisons (e.g., with musicians). I agree with you in that I specifically pointed out that many people besides computer-science majors are very narrow. This is no justification, however. As the son of an artist, I think I can say that such creative outlets are essential, but they do not and should not make up the totality of one's existence.

Secondly, you accuse me of stressing the evils of the machine world. This is an interpretation of my essay that is purely yours. I attempted to make clear that I thought computers could be highly useful tools, but that much of their valuable potential was being lost due to the narrow approach to them. It is a HUMAN failure and not a mechanical one.

FROM: E. ERNEST
TO: G. GROG, W. WOODY, H. HIMA, B. BERLIN, O.O7, K. KOA, Y. YOUJ, S. SQUASH, J. Q. JOHNSON
SUBJECT: THE HACKER PAPERS

Dear Sir or Madam,

You have contributed NOTHING, repeat NOTHING, to the Hacker Papers. As a hacker, it is your responsibility to defend yourself against this vicious attack on your way of life. Do you intend to stand idly by while mealy-mouthed, bleeding-heart, fuzzy-studies majors shred you? Surely not! Voice your opinion! Stand up for your right to enjoy not enough sleep, not enough terminals, and not enough core! What people do with consenting computers in private is their business!

FROM: J. Q. JOHNSON
TO: E. ERNEST
SUBJECT: THE HACKER PAPERS

What, pray tell, are the "Hacker Papers"? A work by Margaret Hacker? A lurid tale of ax murder? A discussion of the psychology of computer nonprogramming? If the last, I have nothing to add, and am not sure I appreciate being addressed as a hacker; I'll have you know that I AM a "mealy-mouthed, bleeding-heart, fuzzy-studies major," since my degree is in political science, which (next to AI) is the fuzziest of studies.

My feelings are aptly summarized in Joseph Weizenbaum's *Computer Power and Human Reason*, where he talks about hackers who become so obsessed in tinkering with their grandiose programs that they lose sight of the problem to be solved. Though I disagree with his ultimate conclusion that the computer cannot be taught to reason like human beings, I tend to agree with his diagnosis of the effect they have on people's ability to reason. I most heartily recommend the book for your consideration.

HACKER GLOSSARY

Hacker: A person who makes constant small and unimportant modifications to a computer program for enjoyment rather than to improve the program.

Hacker display/Display hack: An alteration in a computer program, done sloppily, and, often, for fun. The kind of program involved is for display of information on a terminal screen.

PASCAL: A relatively new computer language (like BASIC or FORTRAN).

Wheel Wars: Wheels are computer users who obtain special privileges. Their wars are games played on computers in which they do vicious things to one another, like blocking other wheels' access to the terminals or trying to confuse and frustrate them. (The wars occur only during late night hours or in summer, when they won't interfere with the work of other programmers.)

AI: Artificial Intelligence, the study of how computers can be programmed to simulate various intellectual processes of human beings, such as language use.

Joseph Weizenbaum: A well-known AI researcher at MIT who condemned hackers in his book, *Computer Power and Human Reason*. Compulsive programming, Weizenbaum wrote, "represents a psychopathology that is far less ambiguous than, say, the milder forms of schizophrenia or paranoia."

HRROI, STI: Esoteric machine instructions used mainly by wheels and computer repair people.

Exec: The executive command processor, the program that carries out the user's orders.

Core: The work space available in the computer, its internal memory capability.

Most of the contributors to "The Hacker Papers" are either still Stanford undergraduates or are working as computer specialists. In order of their messages, they include Kenneth Peter (G. GANDALF), Ernest Adams (E. ERNEST), Robert Kanefsky (K. KANEF), Mark Crispin (M. MRC), Steve Kudlac (K. KUDDLES), and John Q. Johnson (J. Q. JOHNSON). A. ANONYMOUS stands for a hacker who did not wish to be identified. Credit for discovering the papers goes to psychologist Philip Zimbardo, whose commentary follows.

7 Resources

In this final section an attempt has been made to provide information about various types of resources that may be useful in getting programs started and keeping up to date on new developments.

Popular magazines and newsletters have been listed as well as a selected list of books. Names and addresses of software vendors and microcomputer manufacturers are included.

An article on educational software by David Lubar reviews programs from seven software vendors. The information presented is useful in selecting software appropriate for gifted and talented programs.

Finally, a list of organizations that help educators use microcomputers in teaching and learning is provided. This document was prepared by The Resource and Referral Service, National Center for Research in Vocational Education. In all, 10 organizations are described. Included are their address, telephone number, and the name of the person to contact.

Educational Software

David Lubar

Almost any use of the computer can be educational, even when instruction is not the main intent of the program. This tends to turn a review of educational software into a complex task of selection (in itself an educational experience). To narrow the field, the following types of programs will be considered: 1. Those labeled "Educational" by their manufacturers. 2. Those which, while not labeled "Educational," do provide the user with new concepts, new information, or new approaches to problem solving. With these criteria established, but not inflexibly fixed, we'll look at a variety of educational software for home computers.

Almost any use of the computer can be educational, even when instruction is not the main intent of the program.

Edu-Ware

Edu-Pak 1 from Edu-Ware ($39.95) is a disk for the 48K Apple II and Apple II-plus, requiring Applesoft in ROM. The disk contains five programs: "Compu-Read," "Perception," (three programs) and "Statistics." Each program allows for several options and variations, thus creating a large software library on a single disk.

"Compu-Read," designed to improve a reader's speed and retention, begins with a choice of six different programs. "Compu-Read I" places three random letters on the screen for a brief moment. The user must type these letters after they have vanished. If he succeeds, the next set of letters remains on the screen for a shorter period of time. If he is wrong, the time of display increases. At the end, the time of display is shown, as well as the number of letters per second for both the start and end of the segment. This information helps the user gauge his progress.

"Compu-Read II" uses words instead of random letters. At this stage, the skills developed in the first program are called into service. Skills related to recognition are also emphasized and strengthened in this exercise. "Compu-Read III" displays a word on the left and four words on the right. One of the four is either a synonym or antonym for the word on the left. Once again, the words do not remain on the screen for long. The user must type the correct synonym or antonym. This involves both recognition of the words and comprehension of their relationships. The system can train a person to absorb and analyze data in a rapid manner.

"Compu-Read IV" presents a sentence, then asks the reader a question concerning the sentence. The question is always about either the subject or object, thus training the user to scan quickly for information. This technique can greatly increase reading speed. The next two programs are file builders which allow the creation of new word lists for the second and third programs. The series is well developed and seems designed to build up reading skill in discrete segments. There are options to specify the number of trials and the length of time for display. Complete statistics are given after a round, breaking the performance into several factors such as percent correct, display time of first word, display time of last word and rate of letters per second.

Next on the disk is the "Perception" series. These three programs, in high-resolution graphics, test visual perception and the ability to judge spatial relations. "Perception I" concerns lengths of lines and gives a choice of 5 tests. In each, a line must be matched to a specific, illustrated length. The line is controlled with the paddles. The options include two vertical lines on the floor of a room and two crossing horizontal and vertical lines on the rear wall of a room. Anyone who is familiar with optical illusions will realize that finding a match is not always easy. The program responds to the user's guess with the percentage of error in the estimate.

"Perception II" deals with shapes. Again, there are a large variety of options. Basically, a shape with from three to eight vertices (user selects this number) is shown in sections as a window scrolls past it. The player must pick a matching shape. In "Perception III" the match must be made on the basis of size. The player selects from a choice of seeing the shape on a blank screen or against a scale which allows comparison. He also chooses from three ways of seeing the master and test shapes. The number of vertices in the shape and the time it is displayed are also controlled by the user.

Anyone familiar with optical illusions will realize that finding a match is not always easy.

These programs develop not only spatial perceptions but also concentration. And they are fun. With all the options and variations, anyone could use the "Perception" series for a long time without tiring of it.

The last selection on the disk is "Statistics." This contains six programs, including "Chi Square Distribution,"

From *Creative Computing*, Volume 6, Number 9, pages 64–72. Copyright © 1980 by Creative Computing. All rights reserved. Reprinted with permission.

"Mean, Variance, and Standard Deviation," and "T-Test."

The disk is well done. Each program is menu driven, and comes with complete instructions. A lot of thought went into *Edu-Pak I*. It would make an excellent addition to the software library of any school, and could be used by students of almost any age. The programs mentioned above are also available individually on cassette and disk. Edu-Ware has many other educational programs (as well as some simulation programs) on the market and under development, and is also working on new versions of present programs. This process of revision insures a dynamic product. They combine talented programming with skilled educational techniques; the results are good.

Steketee Educational Software

Cassette TO 3 for the 16K TRS-80 ($9.95 + $1.00 p&h), from Steketee's *EDU-SOFT* series, contains two programs which can be used both in the classroom and at home, "Plot," and "Guess the Rule." "Plot" allows the graphing of single or simultaneous equations. Anything within the mathematical capability of the TRS-80, from a simple Y=X+2 to a complex Y=SIN(X)/X*SOR(X), can be used. An equation is entered by being placed in memory as line 400. A second equation can be inserted as line 500 if a graph of simultaneous equations is desired. Since these equations become part of the program, TRS-80 conventions for math symbols must be followed. Once an equation is entered, the user has a choice of either Cartesian or Trig coordinates, as well as a choice of any desired endpoints. If the selected endpoints are too small, the line won't appear on the screen. If this happens, larger boundaries are needed. After the function is graphed, it can be replotted with different endpoints, saved to be combined with the next equation, or deleted from memory. "Plot," which gives good visual representations of many concepts in Algebra and Trig, could be of value to almost any age group.

Both beginners and old pros will be fascinated by the internal view of a computer in action.

In "Guess the Rule," the computer selects an equation, ranging in difficulty from simple linear ones of the form Y=X+a to quadratics such as Y=aX + bX + c. The player is then shown two pairs of X,Y values. Using this information, he must determine the rule, either by guessing the equation or by supplying a correct pair of values for X and Y. Ten equations are presented in each contest.

Another cassette, *TO 4* (same price and configuration as *TO 3)*, contains "Computa-Doodle" and "Simulated Computer." "Computa-Doodle," as the title suggests, is a graphics utility. The left, right, up and down arrows control a cursor which draws lines. An arrow followed by a number will give a longer line. Left arrow followed by "9," for example, will move the line nine spaces to the left, plus one for the move registered with the initial press of the arrow. The "Q," "W," "A," and "S" keys allow diagonal moves, with numbers following to give the degree of slope. Once a diagonal has been started, the slope can be changed just by pressing a different number. This allows smooth curves to be drawn. There is also a command which displays the numbers that represent the graphics in memory, and a command which moves the entire drawing on the screen. "Computa-Doodle" is well designed and easy to use.

"Simulated Computer" is an excellent program. It turns the TRS-80 into a microprocessor. Twenty memory locations (little boxes) are displayed on the screen, along with boxes for input, output, accumulator, program counter and instruction register. A group of three digit commands, for such functions as "add," "subtract," and "skip," is used to program this simulation of a computer. Once a program is entered, the operation of this "central processing unit" is graphically displayed as the user sees memory contents change and as input and output appear in the boxes. There are also modes for slowing the program, and for running in single steps. The program is a great introduction to the concept of a microprocessor. Both beginners and old pros will be fascinated by this internal view of a computer in action. "Simulated Computer" is a good first step on the way to understanding a Z-80 or a 6502 microprocessor.

Steketee programs come with good documentation, including complete instructions and suggestions for using the programs. The programs are well designed for ease of operation and contain many error traps. This cassette can definitely be used in the classroom. Individual users can also enjoy this tape. Any of Steketee's tapes can be ordered on disk for an additional $5.00.

Basics and Beyond, Inc.

Microcosm I for the Level II 16K TRS-80 ($19.95) contains thirty programs on two cassettes. The programs range from games and simulations to utilities and educational aids. The games are nice, though some are reminiscent of previously published ones. Several of the games could be considered educational. "South Pole" allows one or two players to try to reach the pole and return. The players decide how many men and dogs to bring, and how much food and fuel to take along. Daily reports are given as the players pick their routes toward the pole. Aside from a rather long wait while data is being displayed, the game is fun. More varied and complex is "Atlantis." Here, the player must decide on a course of action to save this mythical island from impending doom. He can try to build a dome for protection against the volcano, work toward evacuation, or try one of several other approaches. Many decisions are involved, with many possible outcomes. This is a nice simulation.

Then there are the educational programs. "Country Guess" has the player choose a country. The TRS-80 asks questions until it is able to name the country. This requires the player to know (or learn) a fair amount of Geography; otherwise he won't be able to answer correctly. "Math Table Drill" allows the user to select the number he wishes to study. The computer than presents problems in basic arithmetic which involve the number. This could be a good way to practice multiplication tables. "Spelling Drill" flashes a word on the screen, then waits for the user to spell the word. Any mistake will immediately end the attempt, but another chance will be given. "Spelling Review" allows the user to enter his own list of words for review. There are three levels, each presenting the word for a different period of time.

Microcosm I also contains data-base type programs such as "Flowering Houseplants," "Calories-Food," and "Calories-Ingredients," as well as programs for balancing checkbooks and checking memory. At this price, it is a worthwhile purchase. Included with the cassettes is a 24 page booklet which gives detailed rules for the games and background on the educational programs. The tapes and booklet are attractively packaged in a sturdy plastic binder.

Educational Activities, Inc.

This company markets several programs for the PET and TRS-80. Apple II-plus versions are also in the works. *Missing Facts* ($29.95) can be used by a student with no help from a teacher; the instructions are clear and simple. As in "Math Table Drill" (see above), problems are presented with one missing factor — 5+?=9 — and the user has to supply this missing number. After a correct answer, the whole problem is displayed again. When the guess is incorrect, three chances are given before the computer provides the answer.

Flash Spelling ($14.95) presents words one letter at a time in large script. The whole word remains on the screen for an instant, then vanishes. At this point, the user must correctly spell the word. While there might seem to be no challenge in spelling a word that has just been on the screen, the benefit of this program is that it reinforces learning. By seeing the word one letter at a time, the student is more likely to remember the correct spelling.

"South Pole" allows one or two players to try to reach the pole and return.

Scrambled Letters ($14.95) is for two players, who take turns trying to solve jumbled words. If a player is correct, his opponent gets a new word; on an incorrect guess, the other player gets a shot at the same word. If they both miss twice, the program shows them the word. Extra points are given for speed in this contest. There is only one small problem. Some words, such as "ocean," have anagrams. A player who responded to "aceno" with the answer "canoe" would be told he was wrong. Aside from this, the program is well designed.

Introduction to Mathematics on the Computer ($29.95) is a marvelous, wide-ranging program. It presents the student with math problems at a specific selected level of difficulty. If the student does well, the level increases. The value of the program becomes apparent when the student makes a mistake. After several tries, if the answer hasn't been found, the machine restates the question as a word problem. Instead of using "5+3," for example, it says, "Maybe this way would be easier for you: If I had 3 pencils and you gave me 5 more, how many pencils would I have?" The program is also extremely patient when trying to get answers to yes/no questions during the inital set-up. This is a nice touch, and shows the care that went into the package.

Programs from Atari represent a good concept which has produced fair to excellent applications.

Most of the EA programs make extensive use of large-size letters. The documentation consists of only an insert in the cassettes, but it provides information on changing the data bases for the spelling games, and advice on what to do when problems are encountered. Besides, the programs contain everything the user would need to know. These tapes could be of value in the classroom, and could also be used by students who want to learn on their own, or who need extra help with a subject. This is definitely a quality product line.

Image Computer Products

Now that the Atari home computer has been on the market for a while, other companies are beginning to produce software for it. Image has brought out a nice cassette, *Skill Builder I* ($19.95), containing two educational programs for younger persons. Running on either the 800 or 400, "Number Hunt" has the player move from the center to the edges of a three-by-three grid, trying to find the number that matches the answer to a problem shown at the bottom of the screen. At first, the problems are very simple. If the player does well, the problems become more difficult. A single player can use the program, or two players can compete, trying to be the first to find the number. In the two player version, each player has his own grid. The control is through joysticks.

The same cassette also contains "Bingo Duel." In this game, numbers must be found on a five-by-five grid. Two players can compete, each getting a different level of problems but using the same grid, or the game can be used by a single player. Both games are well explained in the booklet accompanying the cassettes. These programs could be used by children who are learning their numbers or who are learning elementary addition and subtraction. Older children might also enjoy the competitive aspects of the games.

Atari

The *Talk and Teach* programs from Atari represent a good concept which has produced fair to excellent applications. Using the Educational System Master Cartridge, the machine comes on with simple instructions for loading the cassette. After the program is in memory, the computer controls the tape, which gives audio output through the television to supplement the information on the screen. For some applications this is a nice idea. The Sociology set, with sixteen programs on four tapes, is well done. The first program introduces the topic, explains what will be covered, and begins to give a background of basic concepts. As text is displayed on the screen, a narrator repeats the material. Since speech can be faster than reading, the voice is able to give extra information.

Throughout the program, the tape stops and a question appears on the screen, along with two or three possible answers. If the wrong answer is selected, there is a buzz. When the right one is found, the tape continues, often making a comment about the answer.

In the same series is a set of History tapes. These begin with the Greeks and move chronologically forward. The history lessons stress a cause-and-effect approach. Obviously the tapes took the efforts of three professionals; a programmer, an educator and a trained speaker. The lessons are put together with great care for detail, accuracy and interest. The use of text and graphics is well done, and learning from these tapes is a painless endeavor.

There are times when something is used because it is available, not because it is the best thing to use. This could be the case with the *Great Classics* selection of the *Talk and Teach* series. Each story is presented in synopsis with a style reminiscent of *Classics Illustrated.* The program pauses to ask questions, insuring that the reader grasps the plot line, but that isn't enough. The value of literature comes not from the plots, which are often ancient and borrowed, but from the way words are put together. None of the works in this sixteen-story collection, from *Julius Caesar* to *A Tale of Two Cities,* can be viewed as anything but story line. Each work is a classic because of the style and art of the writer. These tapes are pale images of the classics. If they interest a student enough to get him to read the originals, fine; but that doesn't seem likely.

Also from Atari is *States and Capitols* ($14.95). A high-resolution map of the USA is displayed. One at a time, the states are outlined. The player has to guess the state. If he is wrong, the program will tell him the answer. After that, he has to guess the capitol. Again, the correct information is provided after each guess. The game is nicely designed, but would be best used in a classroom, since individuals who used it would quickly learn the states and have no more need of the program.

Creative Computing
(Sensational Software Division)

This review would not be complete without mention of some of the newer educational programs from *Creative Computing*. First, for the TRS-80, is *Ecological Simulations - 2* ($14.95). This contains four programs: "Pollute," "Rats," "Malaria," and "Diet." In "Rats," the player tries to control a population of rats, using various poisons. He can select the length of time between reports, as well as the length of each application and weight of the poison. Trying different methods, the player learns the results of combination programs using varying amounts of poisons.

"Malaria" is a varied and interesting program. The player must select from several measures designed to limit the spread of a malaria epidemic. Attention must be split between preventative and curative measures. At the end of a round, the player is given an evaluation of his

The value of literature comes not from the plots, which are often ancient and borrowed, but from the way words are put together.

work, showing how effective each of his measures was. Bit by bit, it is possible to develop a strategy which makes the most effective use of the various measures. This game quickly becomes absorbing.

"Pollute" takes the user's selection of water temperature, kind of waste, rate of dumping, and treatment method, then shows the effects on a body of water. It contains good error traps and other aids. In entering the amount of waste, a reply that is way too large will result in the comment: New York City has a rate of only 12 parts per million per day. The display graphs oxygen against waste in the water. This program, too, is a good learning tool.

The final simulation, "Diet," allows experimentation with various weekly diets, without the risk entailed from actual experiment. A player can learn what a week of milkshakes would do to him, or a week of sprouts and other vegetables. The output tells how much weight would be gained or lost on this diet, as well as the nutritional makeup of the food.

In these programs, a reply of "-1" as input is used whenever the player needs help in answering a question. The documentation includes detailed explanations of all parameters and variables, as well as exercises to get the user started and advice for classroom activities with the program. Also included is information on the assumptions made in the simulation. The manual itself can be an educational tool.

Another tape from *Creative Computing, CAI-2* ($11.95), is for a 16K Apple II with Integer Basic. It contains three programs designed for computer-assisted instruction. "European Maps" begins with a high-resolution display of Europe. The player is given the choice of naming either countries and capitols or just countries, as well as the option of just entering the first letter of the name of the country. During the game, a dot in the center of one of the countries will begin to flash. The player must name this country. If he is wrong, he will be given the same country twice more at other times during the game. This use of graphics for visual learning is nice, and should produce good results.

"Meteor Math," for beginning and intermediate students of math, takes the pain out of arithmetic. The player is informed of a meteor which is falling toward Earth. He must destroy the meteor with his laser cannon. But he has to answer math problems correctly before he can fire the cannon. It takes a lot of shots to destroy the meteor. If the student does well, the level of difficulty of the problems increases. This is a nicely-designed package which makes good use of graphics in presenting math problems.

The tape also contains "Music Composing Aid," which allows the entry of notes, the replay of the notes or of other saved scores, and the editing of compositions. The music, coming from the Apple's own speaker, sounds like an alto recorder. When using the system for composition, each note is played as it is entered. Notes are coded using a simple method. A C below middle C is entered as C. To go an octave higher, the composer would use CC. There is a range C below middle C to C three octaves above middle C. Along with the program is a data tape containing a Bach composition. The program encourages students to learn musical notation so they can transpose their favorite scores from sheet music into computer data. This package offers a nice variety of programs, and would be worth the price for the music routines alone.

Moving away from educational software, we get to those programs that are intended as aids for teachers. Generally, such programs are concerned with organizing data, and with saving time. First, two cassettes from Educational Associates; one very good, one fairly disappointing. The good cassette, *Readability Index* ($9.95), can be useful not just to teachers but also to those who work in children's literature; especially literature for the high-interest low-readability field. Readability gives a guideline for determining what material can be comprehended by what grade level. In the EA program, the user enters three paragraphs; the first, one from the middle, and the last. The program then gives word count, sentence count, the number of letters, the average length of a word, the percentage of the words that are on the Dale readability index, and the readability level. The Dale long list contains words that are in the vocabulary of readers at certain ages. The percentage given is an estimate; an actual check against the list would take too much time. Along with the printout is a chart for finding the grade level that is appropriate to the readability. This program is useful and well set up for ease of operation.

Considering the overall quality of EA software, their *Grade Averager* ($9.95) is a disappointment. The program allows entry of grades, either letter or numeric, for each student in a class. When all the grades have been entered for a student, an average score and letter equivalent are given. At the end of the program, a summary of all names and averages is furnished. So far, no problem. But the program does have flaws. First, once a grade is entered, it is there for good. There is no way to edit mistakes. Any change would entail redoing the whole file for that student. Also, a wrong entry that is a letter other than "A," "B," "C," "D," or "F" is taken as a signal that the entries for that student are finished. It seems that this cassette could create more work than it saves.

A player can learn what a week of milkshakes would do to him, or a week of sprouts and other vegetables.

The *Apple II Gradebook* ($24.95) from *Creative Computing* is a disk-based utility that allows teachers to set up files containing the names of students and their scores. The user first establishes a roster by entering the names of the students. More than one class can be held on a disk. Once a roster is on file, it can be accessed to add new scores, change scores, change existing information, or add information.

A lot of thought seems to have been devoted to making this program easy for the user. After a name has been entered, the computer shows the name on the screen and asks if it is correct. Getting a "yes," the name is put on file. If the name isn't correct, the computer asks for another entry. This method should virtually eliminate user errors.

With names and scores on file, it is possible to get various statistics from the system, such as scores and averages for each student, as well as his deviation from the mean. Another nice touch: the names can be entered in any order. When they are sent to disk, they will be stored alphabetically. This system is very easy to use. Anyone who can type can have the luxury of a computer grade book. The documentation covers use of the system and recovery from any problems that might be encountered (such as accidentally hitting reset).

As should be obvious by now, there is a lot of educational software out there, and the quality seems to be getting better every month. With careful shopping, any school or individual should be able to fill all software needs for a reasonable price. ☐

Vendor Addresses

Atari Inc.
1265 Borregas Ave.
P.O. Box 9027
Sunnyvale, CA 94086

Basics & Beyond, Inc.
Box 10
Amwalk, NY 10501

Creative Computing
P.O. Box 789-M
Morristown, NJ 07960

Educational Activities, Inc.
P.O. Box 392
Freeport, NY 11520

Edu-Ware Services, Inc.
22035 Burbank Blvd. #223
Woodland Hills, CA 91367

Image Computer Products, Inc.
615 Academy Drive
Northbrook, IL 60062

Steketee Educational Software
4639 Spruce St.
Philadelphia, PA 19139

Printed Materials

Magazines/Newsletters

Apple Education News, P.O. Box 20485, San Jose CA 95160.

Byte, 70 Main Street, Peterborough NH 03458.

Computer-Using Educators, Mountain View High School, Mountain View CA 94041.

The Computing Teacher, c/o Computer Center, Eastern Oregon State College, La Grande OR 97850.

Creative Computing, P. O. Box 789-M, Morristown NJ 07960.

EDU, Educational Products Group, Digital Equipment Corp., M15/M40, Maynard MA 01754.

Educational Technology, 140 Sylvan Ave., Englewood Cliffs NJ 07632.

Kilobaud Microcomputing, Pine Street, Peterborough NH 03458.

Recreational Computing, P. O. Box E, 1263 El Camino Real, Menlo Park CA 94025.

Teaching Computer Programming, 1112 Glacial Drive, Minot ND 50701.

T.H.E. Journal, Information Synergy Inc., P. O. Box 992, Acton MA 01720.

Books

Ahl, David H. (Ed.). *102 Basic Games*, 1978. Creative Computing Press, Morristown NJ 07960.

Ahl, David H. *More Basic Computer Games*, 1980. Creative Computing Press, Morristown NJ 07960.

Ball, Marion & Chase, Sylvia. *Be a Computer Literate*, 1977. Creative Computing Press, Morristown NJ 07960.

Billings, Karen, and Morsund, David. *Are You a Computer Literate?* 1979. Dilithium Press, P. O. Box 92, Dept CT, Forest Grove OR 97116.

Computer-Based Education: The Best of ERIC, June 1976—August 1980. Keith A. Hall, Syracuse University Printing, 125 College Place, Syracuse NY 13210.

Computers in Education Resource Handbook, 1976. Department of Computer Science, University of Oregon, Eugene 97403.

Dertouzos, Michael L., and Moses, Joel (Eds.). *The Computer Age: A Twenty-Year View*, 1979. The MIT Press, Cambridge MA 02139.

Doeer, Christine. *Microcomputers and the 3R's: A Guide for Teachers*, 1978. Hayden Book Co., Inc., 50 Essex Street, Rochelle Park NJ 07662.

Dwyer, Thomas, and Critchfield, Margot. *Basic and the Personal Computer*, 1978. Addison-Wesley, Reading MA 01867.

Edwards, Judith B.; Ellis, Antoinette S.; Richardson, Duane, E.; Holznagel, Donald; and Klassen, Daniel. *Computer Applications in Instruction: A Teacher's Guide to Selection and Use*, 1978. Time Share Corp., Hanover NH 03755.

Frederick, Franz J. *Guide to Microcomputers*, 1980. Association for Educational Communications and Technology, 1126 16th Street, N.W., Washington DC 20036.

Graham, Neil. *The Mind Tool: Computers and Their Impact on Society* (2nd Ed.), 1980. West Publishing Company, St. Paul MN.

Harris, Diana (Ed.). *Proceedings of the National Educational Computing Conference*, 1979. Weeg Computing Center, University of Iowa, Iowa City IA 52240.

Illinois Series on Educational Applications of Computers: Computing-Teacher Education Papers, 1979. Department of Secondary Education, 396 Education Building, University of Illinois, Urbana 61820.

Kosel, Marge, and Carlstrom, Geraldine. *Elementary, My Dear Computer*, 1979. Minnesota Educational Computing Consortium, 2520 Broadway Drive, Lauderdale MI 55113.

Morsund, David. *School Administrator's Introduction to Instructional Use of Computers*. International Council for Computers in Education, c/o Computing Center, Eastern Oregon State College, La Grande OR 97850.

Morsund, David. *Teacher's Guide to Computers in the Elementary School*. International Council for Computers in Education, c/o Computing Center, Eastern Oregon State College, La Grande OR 97850.

Nelson, Ted. *Computer Lib/Dream Machines*. The Distributors, 702 South Michigan, South Bend IN 46618.

Papert, Seymour. *Mindstorms: Children, Computers, and Powerful Ideas*, 1980. Basic Books, 10 E. 53rd. St., New York NY 10022.

Poirot, James L. *Computers and Education*. Sterling Swift Publishing Company, P. O. Box 188, Manchaca TX 78652.

Ricketts, Dick (Project Director). *Course Goals in Computer Education K-12*, 1979. Commercial Educational Distributing Services, P. O. Box 8723. Portland OR 97201.

Taylor, Robert P. (Ed.). *The Computer in the School: Tutor, Tool, Tutee*, 1980. Teachers College Press, 1234 Amsterdam Avenue, New York NY 10027.

Thomas, James L. *Microcomputers in the Schools*, 1981. The Oryx Press, Suite 106, 2214 North Central at Encanto, Phoenix AZ 85004.

Weizenbaum, Joseph. *Computer Power and Human Reason*, 1976. W. H. Freeman, 660 Market St., San Francisco CA 94104.

Willis, Jerry. *Peanut Butter and Jelly Guide to Computers*, 1978. Dilithium Press, Forest Grove OR 97116.

Vendors and Manufacturers

Software Vendors

Acorn Software Products, Inc.
634 North Carolina Ave., S.E.
Washington DC

Apple Computer, Inc.
10260 Bandley Drive
Cupertino CA 95014

Apple-cations
21650 West Eleven Mile Rd., Ste. 103
Southfield MI 48076

Bertamax, Inc.
311 West McGraw
Seattle WA 98119

Bluebird's Inc.
2267 23rd Street
Wyandotte MI 48192

Blue Lakes Software
3240 University Avenue
Madison WI 53705

The Bottom Shelf
751 Dekalb Industrial Way
Decatur GA 30033

Cavri Systems, Inc.
26 Trumbull Street
New Haven CT 06511

Charles Mann & Associates
7594 San Remo Trail
Yucca Valley CA 92284

Compak, Inc.
P. O. Box 14852
Austin TX 78761

Computer Systems Design Group
3632 Governor Drive
San Diego CA 92122

Computer Systems International, Inc.
539 Durie Avenue
Closter NJ 07624

Computer Bus. Syst. of MP
1707 View Street
Myrtle Point OR 97458

Conduit
P. O. Box 388
Iowa City IA 52244

Cook Computer Company
1905 Bailey Drive
Marshalltown IA 50158

Cow Bay Computing
P. O. Box 515
Manhasset NY 11030

Creative Computing
P. O. Box 789-M
Morristown NJ 07960

Curser Magazine
Box 550
Goleta CA 93017

Edu-Ware Services Inc.
22035 Burbank Boulevard, #223
Woodland Hills CA 91367

Educational Activities Inc.
1937 Grand Avenue
Baldwin NY 11510

Educational Software Prof., Ltd.
38347 Grand River
Framington Hill MI 48018

Educational Programs
P. O. Box 2345
West Lafayette IN 47960

George Earl
1302 South Gen. McMullen
San Antonio TX 78237

Hartley Software
3268 Coach Lane #A
Kentwood MI 49508

Hayden Bock Company
50 Essex Street
Rochelle Park NJ 07662

Ideatech Company
P. O. Box 62451, Department 7220
Sunnyvale CA 94088

Instant Software
80 Pine Street
Peterborough NH 03458

Interactive Education
2306 Winters Drive, Dept. 16D
Kalamazoo MI 49002

J & S Software
140 Reid Avenue
Port Washington NY 11050

Krell Software
21 Mill Brook Drive
Stony Brook, NY 11790

Kyde Tyme Project
Ted Perry
2331 St. Marks Way
Sacramento CA 95825

Level IV Products, Inc.
322238 Schoolcraft, Suite F4
Livonia MI 48154

Life Software Ltd.
946 Van Horne Avenue
Willowdale Ontario Canada M2J 2T

Malibu Microcomputing
23910A De Ville Way
Malibu CA 90265

Mean
256 North Washington Street
Falls Church VA 22046

Med Systems Software
P. O. Box 2674
Chapel Hill NC 27514

Micro Gnome
5843 Montgomery Road
Elkridge MD 21227

Micro Learning Ware
Box 2134
North Mankato MN 56001

Micro Power and Light Co.
1108 Keystone
13773 North Central Expressway
Dallas TX 75243

Microcomputer Workshops
10 Elizabeth Place
Armonk NY 10504

Microphys Programs
2048 Ford Street
Brooklyn NY 11229

Minnesota Educational Computing
Consortium
2520 Broadway Drive
Lauderdale MN 55113

Monument Computer Service
Village Data Center, Box 603
Joshua Tree CA 92284

Muse Software
330 North Charles Street
Baltimore MD 21201

Oregon Software
2340 SW Canyon Road
Portland OR 97201

Personal Software Inc.
592 Weddell Drive
Sunnyvale CA 94086

The Pet Program Exchange
Box 516
Montgomeryville PA 18936

Program Design Inc.
11 Idar Court
Greenwich CT 06830

Programmer International Inc.
3400 Wilshire Boulevard
Los Angeles CA 90090

Programs for Learning Inc.
P. O. Box 954
New Milford CT 06776

Programmer Software Exchange
2110 North Second Street, POB 199
Cabot AR 72023

Radio Shack Department
Cma-81010161
1300 One Tandy Center
Fort Worth TX 76102

Sliwa Enterprises
P. O. Box 7266
Hampton VA 23666

Steketee Educational Software
4639 Spruce Street
Philadelphia PA 19139

Stirling Swift Publishing Co.
P. O. Box 188
Manchaca TX 78652

Thesis
P. O. Box 147
Garden City MI 48135

TI Source (Texas Instruments)
P. O. Box 191
Rye NY 10580

TYC Software
40 Stuyvesant Manor
Geneseo NY 14454

Microcomputer Manufacturers

APF Electronics, Inc.
Imagination Machine
444 Madison Avenue
New York NY 10022

Apple Computer, Inc.
10260 Bandley Drive
Cupertino CA 95014

Atari
800 or 400
1265 Borregas Avenue
Department C
Sunnyvale CA 94086

Commodore Pet
950 Rittenhouse Road
Norristown PA 19401

Compucolor
Compucolor II Renaissance Machine
P. O. Box 569
Norcross GA 30091

Cromemco, Inc.
280 Bernardo Avenue
Mountain View CA 94043

Exidy Sorcerer
969 Maude Avenue
Sunnyvale CA 94086

Heath Company
Heathkit
Benton Harbor MI 49022

Hewlett-Packard Company
HP-85 Inquiries Manager
1507 Page Mill Road
Palo Alto CA 94304

Intelligent Systems Corporation
(See Compucolor)
Intecolor Drive
225 Technology Park
Atlanta-Norcross GA 30092

Mattell
Intellivision
5150 Rosecrans Avenue
Hawthorne CA 90250

Motorola
5005 East McDowell
Phoenix AZ 85008

North Star
Horizon II
1440-4th Street
Berkeley CA 94710

Ohio Scientific
Challenger Series
1333 South Chillocothe
Aurora OH 44202

Radio Shack
TRS-80
1300 One Tandy Center
Fort Worth TX 76192

Sinclair Research Corporation
Sinclair Microcomputer ZX80
50 Staniford Street
Suite 800
Boston MA 02114

Smoke Signal Broadcasting
31336 Via Colinas
Westlake Village CA 91362

Southwest Technical Products
SWT PC 6800
219 West Rhapsody
San Antonio TX 78216

Synertek
3001 Stender Way
Santa Clara CA 95051

Texas Instruments
TI 994
12501 North Central Expressway
Dallas TX 75222

MINI-LIST OF RESOURCES

Resource and Referral Service
National Center for Research
in Vocational Education
Ohio State University
1960 Kenny Road
Columbus, OH 43210

ORGANIZATIONS HELP EDUCATORS USE MICROCOMPUTERS IN TEACHING AND LEARNING

Each issue of *Educational R&D Report* features a "mini-list" of resources from the Resource and Referral Service at the National Center for Research in Vocational Education, Ohio State University. This issue focuses on resources for using microcomputers in schools and classrooms.

Additional information on this topic can be obtained from the regional educational laboratory in your area or from the Resource and Referral Service at the address below.

Association for Computing Machinery (ACM)
1133 Avenue of the Americas
New York, N.Y. 10036
Contact: Member Services
212/265-6300

ACM is an association for computer scientists, business system specialists, analysts, and social scientists interested in computing and data processing. The purpose of the organization is to advance the science and art of information processing, including computing techniques and appropriate language for general information and processing, storage, retrieval, transmission/communication, and simulation of data.

ACM has produced a publication on the use of computers by schools, titled *Computer Education for Elementary and Secondary Schools* (Order No. 812810, 1981, 92 pp., $7 for members, $10 for nonmembers). It can be purchased from the ACM Order Department, P.O. Box 64145, Baltimore, MD 21264. All orders must be prepaid.

Association for Development of Computer-based Instructional Systems (ADCIS)
ADCIS Headquarters
Computer Center
Western Washington University
Bellingham, WA 98225
Contact: Gordon Hayes, executive secretary
206/676-2860

ADCIS is an international, nonprofit organization representing elementary and secondary school systems, colleges, universities, business, industry, and military and government agencies.

The purposes of ADCIS include advancing the investigation and utilization of computer-based instruction and management; promoting and facilitating the interchange of information, programs and materials; reducing redundant effort among developers; and specifying requirements and priorities of hardware and software development.

The *ADCIS News* is a bimonthly newsletter mailed to all members. A scholarly publication, *The Journal of Computer-Based Instruction*, is published four times a year (free to ADCIS members, $12 to nonmembers). ADCIS operates several special interest groups including ones for Plato users, mini/microcomputer users, elementary/secondary and junior college personnel, and individuals involved in the Interactive Instructional System Presentation and Authority. Membership in ADCIS is $30 a year for individuals who are affiliated with an institutional or associate member, and $40 for those who are not.

Association for Educational Data Systems (AEDS)
1201 16th ST., N.W.
Washington, DC 20036
Contact: Shirley Easterwood, executive director
202/833-4100

AEDS is a professional organization for those interested in the use of computers in education. AEDS produces three quarterly publications: the *AEDS Bulletin*, which focuses on interchapter communications and meeting announcements ($7 a year); the *AEDS Monitor*, which contains short, timely articles on current directions in educational data systems ($15 a year); and the *AEDS Journal*, which features articles illustrating the latest developments in the application of computer technology to all areas of education ($25 a year).

An individual membership ($35) includes subscriptions to all three publications. The special Fall 1979 issue of the *AEDS Journal* covered "Microcomputers: Their Selection and Application in Education"; single copies can be purchased for $10.

Additional activities include an annual international convention, special seminars and workshops that provide opportunities to examine and discuss the latest developments in educational technology, and a computer programming contest for students in grades 7 through 12.

Computertown, USA!
P.O. Box E
Menlo Park, CA 94025
Contact: Ramon Zamora, director
415/327-0541

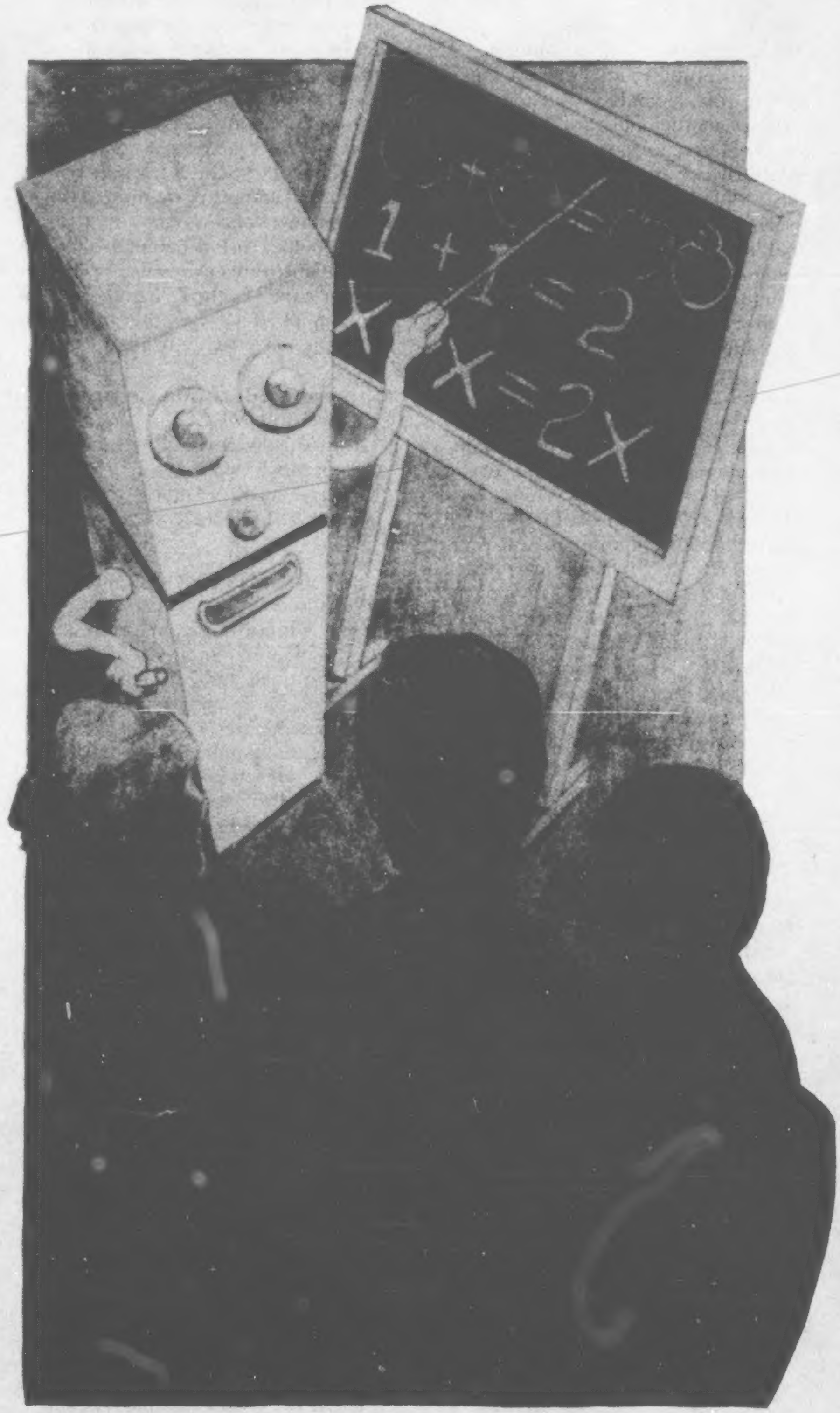
Computertown, USA! is a computer literacy project of the people of Menlo Park, the Menlo Park Library, and the Peoples Computer Company, a non-profit education corporation in Menlo Park. The goals of this National Science Foundation funded project are to develop, test and evaluate a set of course materials and operating procedures for a transportable microcomputer-based community literacy program. The model project emphasizes the use of local community resources.

The courseware is based on a series of six-week classes for children and adults dealing with introductory materials, BASIC programming, and project-based programming experiences. Courseware includes student workbooks, teacher guides, handouts, and lists of recommended software that complement the class activities. These items will be available for dissemination in late 1983.

A free newsletter, *Computertown, USA! News Bulletin*, is published monthly. To obtain a subscription, write to the above address.

CONDUIT
P.O. Box 388
University of Iowa
Iowa City, IA 52244
Contact: Harold Peters, associate director
319/353-5789

CONDUIT is a source for computer-based instructional materials that are reviewed, documented, programmed for ease of transfer, and continually updated. Programs are written in BASIC and FORTRAN. Materials programmed in BASIC will run on most



minicomputer systems and some microcomputer systems. Other CONDUIT materials are being translated to run on the more popular minicomputers.

Microcomputer materials are currently available in biology, chemistry, physics, mathematics and statistics, Spanish, psychology, and sociology. Materials are available for use on the PET 2001, Apple II, and TRS-80.

Three types of informational materials are also available from CONDUIT: general information about the use of computers in higher education; aids for authors and developers of computer-based instructional materials; and reports on CONDUIT reviewed and tested materials. *CONDUIT Pipeline* ($15 a year) is a semiannual publication featuring ideas for use in higher education and complete descriptions of CONDUIT's latest reviewed and tested materials. Aids available for developing instructional computing materials include *CONDUIT Author's Guide: How to Design, Develop and Package Instructional Materials for Widespread Distribution* (1978, $10) and *CONDUIT BASIC Guide* (1979, $10). A publication titled *Microcomputer Versions of CONDUIT Reviewed and Tested Packages* (single copy free) describes CONDUIT packages implemented on microcomputers. A set of guidelines for authors of microcomputer materials is also under development.

International Council for Computers in Education (ICEE)
Department of Computer and Information Science
University of Oregon
Eugene, OR 97403
Contact: David Moursund
503/686-4405

The International Council for Computers in Education (ICEE) is a nonprofit professional organization dedicated to improving the instructional use of computers. *The Computer Teacher* ($14.50 for nine issues per year) is a journal for people interested in the instructional use of computers at the precollege level. It deals with the use of computers and their impact on the general curriculum.

Two instructional booklets are available. *The School Administrator's Introduction to Instructional Use of Computers* is intended for school administrators and school board members who have had little experience with computers. The booklet provides a series of questions and answers about the instructional use of computers and the potential for their use in schools. *The Teacher's Guide to Computers in the Elementary School* can help elementary school teachers with little formal computer training to gain an initial level of computer literacy. (Each booklet costs $2.50 prepaid; quantity discounts are available for 11 or more copies of the same title.) Additional booklets on other topics of interest to teachers are being prepared.

Minnesota Education Computing Consortium (MECC)
2520 Broadway Drive
St. Paul, MN 55113
Contact: Executive Director
612/376-1122

MECC is an organization created to coordinate and provide computer services to students, teachers and educational administrators in Minnesota. At present, approximately 2,000 computer terminals and 2,500 microcomputers are located in schools and colleges across the state.

A Documentation Center makes available written materials ranging from periodic newsletters to curriculum guides that offer assistance in implementing computer use in the classroom. The bimonthly newsletter, called *USERS*, lists available diskettes that can be used with the Apple II, support booklets for these diskettes, and programming and reference manuals. (Non-Minnesota customers must pay a surcharge to cover the MECC operating costs paid by MECC-member institutions.) Both *USERS* and *Dataline*, another bimonthly MECC newsletter, are free upon request.

Northwest Regional Educational Laboratory Microcomputer Software and Information for Teachers (MicroSIFT)
300 S.W. Sixth Ave.
Portland, OR 97204
Contact: Jeri Marler, program assistant
503/248-6800, ext. 437.

MicroSIFT, a project of Northwest Regional Educational Laboratory, serves as a clearinghouse for microcomputer K-12 instructional software information. The clearinghouse focuses both on establishing effective procedures for the collection, evaluation, and dissemination of materials and information, and on providing user support and technical assistance. Additional information is available through *MicroSIFT News*, a quarterly newsletter. To be placed on the *MicroSIFT News* mailing list, write to the above address.

Society for Applied Learning Technology (SALT)
50 Culpepper St.
Warrenton, VA 22186
Contact: Raymond Fox, president
703/347-0055

SALT is a nonprofit organization for professionals in the area of instructional technology (membership, $20 a year).

Members receive the quarterly *Newsletter*. Other publications include the *Journal of Educational Technology Systems* ($22.50 a year for members; $51 a year for nonmembers), *Microcomputers in Education and Training* (1979, $30) and *Videodisc, Microcomputer and Learning System Technologies in Public, Vocational, Medical and Handicapped Education and Training* (1980, $40).

Technical Education Research Centers (TERC)
Computer Resource Center
8 Eliot St.
Cambridge, MA 02138
Contact: Daniel Watt, director
617/547-3890

The Computer Resource Center (CRC) of TERC contains microcomputer hardware, software, curricula and technical information of interest to educators.

CRC maintains representative computers, collects and evaluates machine-specific educational software, and houses a library of related technical and educational publications. *Hands On!* is an occasional newsletter that is free upon request.

CRC also conducts one and two day workshops on educational uses of microcomputers at TERC and at schools in the New England area. Contact the director for additional information.